Doctoral Thesis

Tagging and Smart Textiles

A Contextual Perspective on Constructionist Learning Environments

by

Milena Reichel

Supervisors:

Prof. Dr. Heidi Schelhowe
Prof. Christoph Lischka

Submitted to fulfill the requirements for a degree of
“Doktorin der Ingenieurwissenschaften”
- Dr. Ing. -
at Fachbereich 3 (Mathematik & Informatik)
Universität Bremen

December 6, 2008

Abstract

This thesis seeks to improve the individual's and also the shared creation of meaning in *constructionist* learning environments and suggests as well as develops the approach *constructionism in context* as a method to design such environments. Within the thesis a wide notion of *context* rooted in *phenomenology* and inspired by current design frameworks such as *embodied interaction* are used. This thesis contributes through adding a contextual perspective to the design of constructionist learning environments. As a showcase of an implementation of the concept a *construction kit* for *smart textiles* and *wearables* is discussed.

Zusammenfassung

Hauptfrage dieser Arbeit ist, wie das Konstruieren persönlicher Bedeutung sowie der Austausch von Bedeutung in konstruktionistischen Lernumgebungen unterstützt werden können. Zu diesem Zweck werden eine kontextbasierte Perspektive sowie ein Design-Konzept speziell für konstruktionistische Lernumgebungen entwickelt und dieses beispielhaft anhand eines Construction Kits für smarte Textilien implementiert. Der Begriff von Kontext geht dabei auf eine phänomenologische Tradition zurück und greift aktuelle Design-Konzepte, wie das Embodied Interaction Framework von Dourish, auf.

Acknowledgments

First of all, I would like to thank my advisors and my thesis committee.

It was a pleasure and privilege to work with and learn from Heidi Schelhowe, who supported my work from the very beginning to the very end, and to be part of her research group DiMeB. I admire Heidi's multifaceted work, her excellence in gathering interesting researchers from all over the world, and in realizing visions, as she did with launching DiMeB's workshop program.

Christoph Lischka's outstanding knowledge and interest in many fields, including philosophy, programming languages, physical computing and bio arts, was a great inspiration. Christoph has provided ongoing invaluable support especially on the theoretical parts of my thesis which I am very grateful for.

During my time at the DiMeB research group and at the graduate school "Advances in Digital Media" I had the opportunity to work with many people from very different backgrounds and with diverse foci. I would like to thank all of my colleagues—past and present of both groups, especially Nadine Dittert, Torsten Grüter, Eva-Sophie Katterfeldt, Anja Kümmel, Irmgard Laumann, Anja Osterloh and Gerald Volkmann.

Lena Berglin, Leah Buechley and Fred Martin shared their knowledge and experiences on smart textiles and working with children and young people with me and I learned a lot from them. The Arduino core development team developed the great Arduino software and hardware and they were always interested in my work and had helpful suggestions. It was a pleasure and a great inspiration to work with our partners in the EduWear project, namely with Deirdre Butler, Eimear O' Callaghan, Marion Ellwanger, Siw Eriksson, Javier Ferreira Gonzalez, Nick Hine, Tünde Kallai, Tomas Molnar and Magda Nagy-Czirok.

Some undergraduate and graduate students supported me as co-workers or with their diploma theses. To name only a few, I would like to thank Carolin Behrens, Lutz Dickmann, Daniel Gehrke, Kristian Gohlke and Stephan Plassmeier.

This thesis would not have been possible without support from the European Commission, who granted the EduWear project, and without support from the Klaus Tschira Foundation and the graduate school "Advances in Digital Media".

My gratitude also goes to all children and young people that took part in our workshops (and their parents). Without them, this research would not have been feasible.

Finally, I would like to thank my friends and family, especially Daniel Kalinke, Roland Knauff and Hanjo Meyer-Rieke who supported me over the (not always easy) last years.

Contents

1 Introduction: Situating the Thesis **1**

1.1 Personal Motivation . . . 3
1.2 Research Goals and Questions . . . 4
1.3 Methods . . . 5
1.4 Overview . . . 6

2 The Interaction Perspective: Interaction Design and Context **9**

2.1 Interaction Design . . . 9
2.2 Interaction in Context . . . 10
2.2.1 Different Notions of Context and Creation of Meaning . . . 11
2.2.2 Theoretical Underpinnings: Phenomenology . . . 12
2.2.3 Theoretical Concepts: Life-world . . . 13
2.2.4 Theoretical Concepts: Creation of Meaning . . . 14
2.3 Towards Design: Embodied Interaction . . . 15
2.3.1 Design Concepts: Design Process . . . 17
2.3.2 Design Concepts: Familiarities and Affordances . . . 18
2.3.3 Design Concepts: Flexibility and Appropriation . . . 18
2.3.4 Design Concepts: Dourish's Design Principles . . . 19
2.4 Summary and Implications . . . 19

3 The Learning Perspective: Constructionism **23**

3.1 Theoretical Concepts: Papert's Constructionism . . . 24
3.2 Theoretical Concepts: Papert's Samba Schools and Beyond . . . 26
3.3 Theoretical Concepts: Vygotsky's Social Constructionism . . . 29
3.4 Theoretical Concepts: Social Constructionism Inspired by Papert and Vygotsky . . . 30
3.5 Theoretical Concepts: Constructionist Culture . . . 31
3.6 Theoretical Concepts: Distributed Constructionism . . . 32
3.7 Theoretical Concepts: Summary . . . 32
3.8 Design Concepts: Programming in Constructionist Learning Environments 34
3.8.1 Design Concepts: Playful Learning and Hard Fun . . . 34
3.8.2 Programming and the Role of Programming . . . 35
3.8.3 Design Example: The Logo Family . . . 38

3.8.4 Design Example: The Smalltalk Family . . . 47
3.8.5 Design Example: MUDs and MOOs . . . 50
3.9 Design Concepts: Design Process . . . 52
3.10 Design: Summary . . . 53

4 Towards a Concept: Bringing Together Context and Constructionism 55
4.1 Context and Constructionism . . . 55
4.2 Designing Contextual Constructionist Learning Environments . . . 57
4.2.1 Familiarities . . . 57
4.2.2 Flexibility and Appropriations . . . 58
4.3 Principles . . . 58
4.4 Relation to Our Vignette . . . 61

5 Suitable Application Domains: Tagging and Smart Textiles 63
5.1 Social Software . . . 64
5.1.1 Children's Use of the Internet, Tagging and Mobile Devices . . . 65
5.1.2 Social Navigation . . . 67
5.1.3 Tagging . . . 69
5.1.4 Meaning Creation and Sharing of Meaning in Tagging Systems . . . 77
5.1.5 Studies of Existing Tagging Systems . . . 80
5.1.6 Conclusion . . . 88
5.2 Smart Textiles . . . 91
5.2.1 Classifications of Smart Textiles . . . 93
5.2.2 Materials . . . 93
5.2.3 Techniques . . . 96
5.2.4 Applications and Projects . . . 97
5.2.5 Context and Creation and Sharing of Meaning . . . 105
5.3 Participatory Software Design with Children . . . 111
5.4 Summary and Conclusion . . . 113

6 Designing the Learning Environment: EduWear and Amici 115
6.1 The Design Process . . . 116
6.1.1 Design Workshops Structure and Approach . . . 116
6.1.2 Contextual Inquiry . . . 117
6.1.3 Future Design Session . . . 119
6.1.4 Results . . . 119
6.1.5 Ongoing Design Process . . . 120
6.2 Observation: Interaction in the Real-World . . . 120
6.3 The EduWear Construction Kit . . . 122
6.3.1 Concept and Iterations . . . 123
6.3.2 Implementation and Technical Details . . . 124
6.3.3 Results from Observing Interactions with the Prototypes . . . 134
6.3.4 Conclusion . . . 136
6.4 Programming Environment . . . 137

6.4.1 The Concept . . . 137
6.4.2 Technical Details and Implementation . . . 140
6.5 Conclusion and Relation to Constructionism in Context . . . 152

7 Evaluation: Piloting and Testing the Prototypes 155
7.1 Usability . . . 156
7.1.1 Quantitative Data . . . 157
7.1.2 Qualitative Data . . . 158
7.2 Learning Experiences . . . 162
7.2.1 Technology Comes Alive . . . 163
7.2.2 Case Study: “Lego Is Stuff for Children, for Babies” . . . 165
7.2.3 Case Study: “You Wanna Build a Dinosaur and out Comes a Tank” . 169
7.2.4 Smart Fashion . . . 171
7.2.5 Case Study: “It Looks Really Cool” . . . 172
7.2.6 Case Study: “Wow, I Can Make Them Blink” . . . 175
7.2.7 Categorization of the Outcomes . . . 178
7.3 Gender . . . 182
7.4 The Tagging Process . . . 182
7.4.1 Tag-based Recommendations . . . 186
7.4.2 Usage Statistics . . . 187
7.5 Conclusion and Summary of the Evaluation Results . . . 187

8 Results and Conclusion: Reconsidering Our Work 189
8.1 Summary . . . 189
8.1.1 Reflection on the Methods . . . 191
8.2 Results and Conclusions . . . 192
8.2.1 General Results and Insights . . . 192
8.2.2 Implications for Constructionism in Context . . . 193
8.3 Contribution . . . 194
8.4 Future Directions and Vision . . . 196

References 212

Figures 215

A Two Example Programs 217
A.1 Loop, Setup and Method . . . 217
A.2 Jacket for the Blind . . . 219

B Questionnaires and Interview Guidelines 225
B.1 Questionnaires from the EduWear Project . . . 225
B.1.1 Questionnaires Before the Workshop . . . 225
B.1.2 Questionnaires After the Workshop . . . 232
B.2 Guidelines Tagging Interview . . . 238

Chapter 1

Introduction: Situating the Thesis

Interaction design is one of the core areas of digital media. How do we as humans interact with a computer of any kind (it can be a mobile device as well as a piece of smart clothing or a desktop computer) and with other humans mediated through the computer? And how can we design for this interaction, so that different people in different environments can use the digital artifact? At the moment, the paradigms are changing, new approaches based on *usability* are called *funability*, *user experience* or *embodied interaction*: interaction design that considers social practices and real-world user experiences.

Audiences of users with special demands for interaction design are children and young people. Children and young people are a very diverse target group. Depending on their age and background they have different abilities. However, compared to adults, one can say that children and young people in general are less patient with technology (especially when dealing with complexity, see for example [Nielsen, 2002]), have limited abilities to express themselves verbally and have different cognitive skills that develop continuously. On the other hand, children and young people are less afraid of trying out things on the computer and have a less tightened opinion how computers should work compared to adults—at least that is what I experienced in working with youngsters. Interaction design for children is a research field that tries to develop technology which suited for young users, but also adapts and develops new evaluation and design methods that can be used with the special target group. Although the research field is quite young, it grew over the past years and is established by now, known through an international ACM conference, the *interaction design for children* (IDC) conference.

Digital media in education is the research field specialized in digital media applications for educational purposes and is closely connected to interaction design for children whenever designing for young learners. Designing learning environments usually happens in relation to a specific learning theory—in our case we base technology design on *constructionism*. Constructionism's basic assumption is that the learner actively constructs knowledge and that the process is most successful when the learner makes a

personally meaningful object (to anticipate Papert's term). We will discuss the theory in more detail in chapter 3. To create digital learning environments for children and young people both theoretical backgrounds matter: We always have to design for an interaction with and through technology, while, on the other hand, the learning theory's assumptions on the learner and learning process will influence the design of the learning environment. With this thesis, we aim at bringing both perspectives closer together in connection to a specific question, namely: How to design learning environments that are closely connected to the user's context.

As an approach within the DiMeB research group[1], we organize and study workshops for children and young people typically between nine and fourteen years old. During the workshops, the participants design and construct technology themselves. The following *vignette* or scenario that is central to this thesis summarizes some problems and observations we made during the work with children and young people. The role of the vignette (see section 1.3 for details on the method) is to serve as a starting point for our investigation and as an example to which we will refer at several times in this thesis.

Vignette

> *Lena, Tobi and Marc have finished building their robot with the help of the manual. Lena is a bit bored; she would have preferred to build something that looks nicer. She did not like the fighting robots from the tutorial. They actually tried to build something else, but the time was short and they were not allowed to glue feathers to the brick. Now the kids are in the process of programming the robot. They want it to go forward till it drives against a wall and then go back. The task seems easy enough, so they start the programming software. "Where is driving? There is 'Action' and 'Variables' but no 'Driving'?" wonders Marc. "Here are also some programs for the Acrobot. Wow, there is one that is called 'dodge' that is exactly what we want." "Does it say more?" asks Lena. "No, only 'dodge'". They try out the program but their robot behaves strangely and falls off the table. Tobi starts to repair it. A tutor explains Marc and Lena that an Acrobot program won't work on their model. They will have to make their own 'dodge program'.*
>
> *"Is there an example how to do it?" asks Lena "Can you show me?" The kids do not understand right away how to create their program. Instead of trying to find a solution they get distracted. "Oh, here it says 'melody'" says Lena. "OK, then let's play a melody instead" answers Marc.*
>
> *Tobi finishes repairing the robot and tries to understand the program the others made. It is not clear to him why they used 'melody' now. Well, he will then stick to the tinkering side and leave programming to the others.*

When reading the scenario some problems catch one's eye immediately. Children and young people seem to prefer "building their own" artifact (their personally meaningful

[1] http://www.dimeb.de

object) instead of building a given design from a manual or textbook, as you can guess from Lena's frustration. Also, programming the artifact is a critical phase of the construction process, as the children and young people often do not have any experiences with programming and do not understand the designer's view on the world, for example categories or names. For an experienced programmer, a category like "operators" is easy to understand. For someone who is supposed to program for the first time, it can be confusing or plain meaningless.

If there is no help from tutors or peers around, children and young people tend to loose their ideas and just program "something". This might result in frustration and in the feeling that they are not able to create technology—the opposite of playful learning and empowerment constructionism emphasizes. From a theoretical point of view, these observations correspond with constructionism that focuses on building a personally meaningful object and with the concept of embodied interaction that points out the importance of creating environments that are "in the world" of the user, as the software used in the scenario is obviously not. We will discuss how the problems relate to our research question in the next section.

These experiences, based on the observations of real workshops with children and young people, brought up the idea to make programming software for children and young people that is more flexible and allows bringing in their own ideas on how to call and categorize elements. This software should be combined with a physical material that is strongly connected to the child's world of everyday life on the one hand, but that is easy to use as a construction material on the other hand.

1.1 Personal Motivation

I chose this particular scenario to show that designing and actually making technologies in a learning context is the topic I am interested in. Not only for children and young people, learning to design technology (e. g. learning how to program, but also learning about circuits and electronics) can be hard. On the other hand, knowing about technology is a great empowerment, because it enables a person to design, implement and alter digital artifacts. I struggled a long time myself, especially with programming, and only through tangible artifacts I finally started to dig deeper into it and learned how much fun[2] programming can be. The *tangibles* offer immediate feedback of the program written in the real-world, and are motivating and easy to use for creative projects. To move a robot or make an LED blink feels—at least for me—like a great success, compared to printing a "Hello World" phrase onto the screen. Tangibles can be integrated into personal interests easily, for example one can create artifacts related to sports or music. Or—highly motivating for me—one can integrate logic into clothing, inventing a new kind of fashion and maybe even wear the garments in everyday life.

The downside of working with tangibles is that debugging can be very hard. Often only little support is available; large parts of the development process pass by while

[2]Programming can be fun, especially *hard fun* as Papert called it. We discuss *playful learning* and *hard fun* in more detail in chapter 3.

searching for solutions on the Internet. Even when using tools that are supposed to make the process easier, little tricks and expert knowledge are required to finally make things work (e. g. you have to cover Lego Mindstorms[3] with a piece of cardboard to make the upload via infrared possible). To find help and support on the Internet, one has to query precisely or ask questions oneself trying to make clear what exactly the problems are. As a precondition, one has to reflect on one's own process, maybe abstract from a concrete example and think about how others may formulate or encounter the problem.

While searching for solutions on the Internet can be quite hard, other easy-to-use technologies and applications have become very popular on the Internet. Technologies that focus on social connections are often called *social software* and facilitate finding other people that share certain interests. Examples of these technologies are *blogs*, *wikis* and *tagging*. These technologies are commonly summarized under the term Web 2.0 [O'Reilly, 2005]. The most important aspect of these "new" applications is *user-created content* and—in the case of tagging—*user-created metadata*. Tagging is the process of labeling any kind of item with one or more free text keywords. In my opinion, tagging is one of the most promising technologies of Web 2.0, as it is a technology applicable to a lot of different scenarios. I developed the idea to apply tagging to a scenario as mentioned before, to integrate the missing flexibility within the software.

The combination of tangibles on the one hand and social software on the other hand seems to be a very powerful connection to further ease the hard task of learning to program. Tangibles provide a physical feedback of programs the user writes, while social software can help to find examples and support and maybe even to find a community for programming and building the kind of artifact the user is interested in.

1.2 Research Goals and Questions

This thesis aims at developing a concept on how to integrate context into constructionist learning environments. The concept can be seen as a method to develop different constructionist learning environments. We also aim at developing and studying an environment that implements our concept. The environment will be a *construction kit* for *smart textiles* and *wearables*. It includes a programming environment to make the textiles *smart* by programming them. The learning environment will use smart textiles and tagging as a way to integrate context into a constructionist learning environment. Our hypothesis is that both application domains (tagging and smart textiles) are very suitable for this integration. The smart textiles have a connection to the young peoples' *life-worlds* (see section 2.2.3 for an explanation of the term), because they know how to deal with clothes and textiles, and because fashion is a very important topic in youth culture. In the programming environment, we use social software features to relate to the young peoples' world and experiences—to the (social) context in which they use the Internet for communication.

[3] http://mindstorms.lego.com/

The problems we illustrated in the vignette can be summarized by saying that technology, as well as the whole learning environment, fails in being meaningful to the young users. It does not consider the context it is used in, when looking at the problem from a perspective that deploys a wide notion of context. The main question we ask in this thesis is how to allow children and young people integrating their context? More detailed questions that we want to work on are:

1. What can we learn from design frameworks—especially from embodied interaction—to make the contextual perspective in learning environments more explicit?
2. How are constructionist learning environments designed to connect to the children and young people's context?
3. Are social software and smart textiles suitable media to connect to the children and young people's life-world?

We can differentiate between the individual's personal creation of meaning in her own context on the one hand, and the social context of a group of people that allows them to share meaning on the other hand. The way we understand it, the first step to share meaning is the personal creation of meaning and the externalization of this meaning. Only then, successful sharing of meaning can take place. This is why we concentrate on the creation and externalization of personal meaning as a precondition for shared creation of meaning.

We see the concept of how to consider context as an important extension of the design of constructionist learning environments. It summarizes different trends that we see in constructionist learning environments as well as in interaction design and makes them explicit. We believe that being aware of this perspective will result in better designs for constructionist learning environments.

While we see the concept as our main contribution, the construction kit that we develop is also a novel approach and technology and is an important part of our contribution.

1.3 Methods

The method we deploy in this thesis is inspired by *vignettenanalyse* or *factorial survey* (see [Backhaus et al., 1989] for details). We started with a scenario, our vignette, in section 1. In the social sciences, a vignette is a scenario that includes different attribute combinations and will be presented to test persons in order to rate these vignettes [Steiner and Atzmüller, 2006]. Another possibility is, to use the vignette to communicate results of a research project by wrapping them in the scenario. The scenario makes the results easier to understand for the reader or another external person. In our case, we use vignettes in a slightly different way. For us, the vignette is a fictional scenario—-that we constructed in order to communicate problems we experienced when working with children and young people—that consists of different aspects of the problem we attempt to solve. We use the vignette throughout the thesis to explain concepts and problems,

as examples, or for illustration purposes. The way we use the vignette is similar to approaches in software development, e. g. modeling persons in the *persona* approach (cf. [Pruitt and Grudin, 2003]).

In addition to our vignette, we start by developing our concept in a hermeneutic analytic approach and draw on theoretical underpinnings from interaction design and approaches based on phenomenology as well as from constructionist learning theory. By bringing them together, we can finally derive and develop our concept.

In our concept, we conclude that we need to know the user's context and social practices around technologies that we take up to design constructionist learning environments. Therefore we study the context by using statistical methods as well as methods from *participatory design*, e. g. *cooperative inquiry* (see for example section 5.3).

When it finally comes to the implementation, we follow Maeda's paradigm, namely: "imagine, realize, critique, reflect, and iterate" [Maeda, 2000]. The "imagine" as well as the "critique" phase are done with our users in a participatory design process and we deploy methods from usability research as well, to support the critique phase.

For the evaluation of our prototype, we use different methods that we introduce and discuss in chapter 7 in more detail.

1.4 Overview

In the second chapter, we introduce one of the basic theoretical concepts we build our approach on. We introduce a wide notion of *context*. The notion we refer to is used in concepts in interaction design that draw on a background in phenomenology. We see context as the world as it reveals itself to the individual and assume each person has her own interpretation of the world. Shared meaning can be constructed through shared experiences and communication. We also introduce design concepts that take up ideas from phenomenology and apply them to the design of digital artifacts. The concepts we introduce are Dourish's embodied interaction framework, as well as—also based on Dourish—*affordances*, *familiarities*, *appropriations* and *flexibility* (for more details see section 2.3.2).

The third chapter introduces constructionist learning theory with its basic assumptions as well as its extensions towards considering the learner's context and focusing on the creation of shared meaning. We then introduce examples of constructionist learning environments—programming environments to be more precise—in a later section of the third chapter, to see, how they implement the theoretical assumptions we discussed before.

From the constructionist background, we can identify a shift towards "sharing creations" and a placement of learning applications into *constructionist communities* sometimes referred to as *social constructionism* to provide a supportive context and shared meaning. This shows that context has always been an important factor in constructionism, but rather implicit.

The learning environments that implement social constructionism do so by taking up topics and domains their target groups are familiar with, e. g. the programming language

Scratch uses video and sound as materials, because these are part of youth culture, by building (real and virtual) communities around the construction activity and by making parts of the creation shareable (in the sense of exchangeable, e. g. via web-based sharing).

We argue that although these approaches solve our problem partly, it is necessary to extend social constructionism with the notion of context towards a wider emergent social context the way we introduced it connected to interaction design. The contextual view has to be expanded to the design of programming environments, and the importance of context has to be made explicit.

In the fourth chapter, we summarize the findings from the chapters on our theoretical underpinnings and lay out how we understand *constructionism in context* and furthermore, what implications this concept can have for the design of constructionist learning environments. We conclude with a set of "design principles".

The fifth chapter introduces the construction part of the thesis. We introduce two application domains, namely tagging and smart textiles. Then we discuss why they are suitable as application domains to design a constructionist learning environment.

The sixth chapter concentrates on the EduWear construction kit including the programming language Amici that combines tagging and smart textiles in a learning environment. The construction kit serves as an example of the implementation of the concept and as an illustration of how the concept can be used. For us, it also provides the possibility to evaluate and refine the concept.

Within the seventh chapter, we evaluate the EduWear prototypes by observing users interacting with the prototypes as well as conducting more formal tests and looking at the actual artifacts the users created in educational workshops using the EduWear construction kit and Amici as programming environment. By comparing individual's experiences in a robotics workshop with Lego Mindstorms technology and a smart textiles workshop we found that the participants of the smart textiles workshop related the material much more to their life-world and also to the world of work than the participants of the robotics workshop.

In the eighth chapter, we discuss the evaluation results and what they mean for our concept. We also discuss future directions and impacts this research may have.

Chapter 2

The Interaction Perspective: Interaction Design and Context

We place this work in the context of interaction design. Many concepts we discuss are referred to as works in the domain of *human computer interaction* (HCI), of which interaction design is a vital part.

First of all, we define basic terms necessary for further discussion. Afterwards, we introduce a new trend we can see in the field of interaction design—a trend towards seeing user and interaction as part of the world and the user's action as dependent on her context. Context is a difficult and slippery notion, therefore we discuss in more detail what we mean by it, relating to a background based on phenomenology and design frameworks, especially based on Dourish [Dourish, 2001b].

2.1 Interaction Design

Interaction design is connected to different theories and approaches. First of all, we would like to start with a definition of the term interaction. We will use Crawford's notion of interaction, because his basic metaphor fits very well with our special focus i. e. on smart textiles. Crawford described an interaction as a dialog or—as he called it—a conversation.

> Interaction: a cyclic process in which two actors alternately listen, think, and speak. [Crawford, 2002, 3]

Crawford emphasized the importance of having two participants in an interaction. For example, according to Crawford a dancer does not interact with the music [Crawford, 2002, 9], because the music does not react to the user's actions (unless of course, we are in one of the novel interactive installations where it does). Crawford also regarded interaction as a continuum with different levels instead of a true or false value. These levels are derived from his basic metaphor—conversation—and are: listing, thinking and speaking.

In an interaction between a human being and a computer, listening would cover the way the user inputs data in a computer. That can be text-based commands in a UNIX shell, but also gestures when wearing a smart wrist band. Thinking refers to what happens after the input. In the shell example, the computer thinks only little—the user has to type in commands correctly, one after another. In the wrist band example, the computer thinks a lot to interpret what the user meant by her gesture. Speaking is the output of the computer. That could be text on screen, but also the temperature of a tangible object that changes.

When taking up Crawford's definition of interaction, *interaction design* would then be the process of designing a conversation. The "better" each aspect is designed, the more interactive the final product would be. A main advantage of digital media for interaction compared to other media, is the media's advantage in thinking (of course thinking refers to machine thinking, not to that of a human).

2.2 Interaction in Context

From our experiences with conversations in our everyday life, we surely noticed that the quality of a conversation not only depends on a common language and shared meaning of the conversation's topic and terms, but as well on the whole context we are in: The surrounding environment with its background noise, our relationship to another person, our roles in society, but also our current mood and others' reactions. These influence and shape how we interpret what happens during a conversation. We will discuss in more detail what we mean by context in this regard in the following sections.

The observation we just described—the dependency of the creation of meaning on a certain context—is a common ground for different approaches in sociology and philosophy and their application to the design and understanding of interactive systems. Some approaches share a perspective, where interaction is embodied in the world and the world is seen as a dynamic complex system with different actors that might be of human, material or whatever nature. All these actors have an agency and the constellation in which meaning creation happens is emergent. Approaches that share these ideas on a philosophical level can be found in phenomenology as well as in contemporary theory, e. g. Haraway [Haraway, 1991] or Latour [Latour, 2007].

From the design perspective, we find Dourish who draws on phenomenology. The basic idea is that a person and her background are inseparably linked. This idea has of course implications for the design and the design process of interactive systems, because it is not possible for the designer to foresee and model an interaction in all aspects. To consider context in the design of computational artifacts is always difficult, because

> programs describe and execute operations on informational objects in a largely decontextualized way. [Floyd, 2002, 27]

There are different philosophical schools and approaches that have been applied to the design of interaction and digital artifacts. We have mentioned approaches rooted in phenomenology or contemporary theory in the last paragraph. Other approaches draw for

example on *analytic philosophy*. We decided to draw on phenomenology at first hand, because the approach is widely used, especially in interaction design (for children) (cf. [Fernaeus et al., 2008], [Sengers and Gaver, 2006], [Coffin, 2008]). With Dourish's embodied interaction framework, we can use an existing approach to design based on the theory. Therefore we rely heavily on Dourish [Dourish, 2001b] as access to phenomenology and secondary literature we base on. We refer to primary sources when we attempt to dig deeper into a certain aspect.

2.2.1 Different Notions of Context and Creation of Meaning

Context plays a great role in interaction design, but also in the line of research manifested in *ubiquitous computing*, *context-aware computing* or *pervasive computing* [Dourish, 2004]. We start our investigation on context with the *Special Issue on Context-Aware Computing of Human-Computer Interaction* [Moran and Dourish, 2001]. Within the publication there is a differentiation between different perspectives on context within the fields we just mentioned: a rather formal one on the one hand and, on the other hand, a human-centered or social one. These opposed views are also called the *positivist* and the *phenomenological* school of thought underlying HCI (cf. [Chalmers, 2003]).

Applying the technical view on context, one will most likely attempt to gather information of the user's environment as data for the computer to process and model the context the user is currently in. The information can be of spatial and temporal nature and might also include information from sensors (for example to gain information about a user's position). After analyzing the user's current situation the computer system will offer services and configurations it considers appropriate to the user's context according to its model. In our robotics example, the technical context could include certain hardware, e. g. a robot model or environmental context e. g. lighting conditions. Dey et al. [Moran and Dourish, 2001] gave the following definition of context from their technical perspective:

> Context: any information that can be used to characterize the situation of entities (i. e. whether a person, place, or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically the location, identity, and state of people, groups, and computational and physical objects. [Dey et al., 2001, 5]

Context in the definiton of Dey et al. referred to a model of the external world with actors mapped as objects and relations between those objects. It did consider the user and even other users she might interact with, but in a fixed model. The context in this notion can include the actions the user is performing, but the actions are not necessarily linked with the creation of meaning.

In this thesis, we will apply the other view on context we already mentioned: the social or human-centered one. Even though it has another focus, it does not exclude the technical context. The human-centered perspective is related to the humanities and their application in interaction design. Many researchers (see for example

[Dourish, 2001a], [Fernaeus et al., 2008] or [Sengers and Gaver, 2006]) regarded Suchman's work on *Plans and Situated Action* [Suchman, 1987] as entry point for an alternative view on context in human computer interaction. Suchman studied people using digital artifacts with *ethnomethodology*. As her main contribution, from these studies she found out that people interact with digital artifacts in a spontaneous manner. Meaning arose in the *ad hoc* situation dependent on the context. The users' actions largely depended on the current situation that emerged. They did not plan their actions beforehand and did not act intentionally most of the times.

The current and emergent context will change the way the user creates meaning and interprets responses of the system. This situated notion of context based on Suchman's work puts action and the experiences of the individual that is acting in the focus. This view draws on the philosophical background of phenomenology sharing for example Husserl's view on creating shared meaning mainly through language. Husserl is a well-known philosopher from the early twentieth century, who is also known as the father of phenomenology.

> ...every occasion of human communication is embedded in, and makes use of, an unarticulated background of experiences and circumstances. Communication in this sense is not a symbolic process that happens to go on in real-world settings, but a real-world activity in which we make use of language to delineate the collective relevance of our shared environment. [Suchman, 1987, 180]

Context is dependent on the action it is associated with. Suchman compares human computer interaction with face-to-face human interaction, assuming the same principles underlay the interaction. That means, when we communicate with another human, we behave differently depending on our mood, the environment and many more factors. That is also true, when interacting with the computer.

2.2.2 Theoretical Underpinnings: Phenomenology

This very brief introduction to phenomenology can be extended by looking e. g. at [Dourish, 2001b]. Phenomenology is a philosophical approach based on Husserl, who got frustrated with the distance between science and the everyday life of people in his work as a mathematician. In phenomenology, the focus is on how an individual experiences the world, a view opposed to that of Husserl's contemporaries.

> Phenomenology, in a general sense, is a practice of reflection upon the structures of lived experience. [Coffin, 2008, 218]

The well-known philosopher Heidegger (who was one of Husserl's students) extended Husserl's phenomenological perspective from asking how the individual *perceives* the world subjectively, to how the person creates meaning through *acting in the world*. Opposed to Husserl's view, for Heidegger the human and her actions in the world belong together—only through actions within the world one can create meaning. For Heidegger,

the world is a place where we act and the world reveals itself to us through our actions. Husserl, on the other hand, argued that the individual relates to the world through perception. In Dourish's words, the main difference in both schools of thought is:

> However, where Heidegger broke with Husserl was in rejecting the mentalistic attitude Husserl has adopted. By 'mentalistic', I mean a focus in Husserl's phenomenology on cognitive, mental phenomena separated from the physical phenomena of mundane existence. [Dourish, 2001b, 107]

By applying Heidegger's perspective, Dourish comes to the conclusion that:

> It is the way in which we act—the practical tasks in which we are engaged, and how they are accommodated into the world—that makes the world meaningful to us. [Dourish, 2001b, 108]

By considering the phenomenological perspective on context, we can conclude that context is not stable and is made through acting in the world and depends on the setting of an action. For an individual, the world opens up and at the same time limits possibilities for action. Heidegger referred to this *field of possibilities* with the term *Lichtung* [Heidegger, 1967, 170] or in the English translation from Macquarrie and Robinson [Heidegger, 1962, 214]: *clearing*.

Phenomenology is a background we can draw on, when investigating how a person creates subjective meaning from acting within the world. For the question on how we then share this meaning among others, we would like to introduce Schütz' work because—according to Dourish:

> Husserl and Heidegger had developed phenomenology in different directions, but they had nonetheless both concentrated on the *individual* experience of the world. [Dourish, 2001b, 110]

2.2.3 Theoretical Concepts: Life-world

Schütz was concerned with the question of *intersubjectivity*: How do people reach a common understanding, a shared meaning? One concept, we would like to introduce, is the *life-world* or *Lebenswelt*. The concept is important for our thesis, because—as we already introduced with our vignette—in our approach we have to connect to children and young people's context. For this context, the life-world plays an important role. Although the term was used before, it moved into the center of attention connected to Husserl's notion of it. According to Dourish, Husserl's life-world is

> the intersubjective, mundane world of background understandings and experiences of the world. [Dourish, 2001b, 106]

He introduced and referred to the world people experience every day without being conscious about it or question it with the term. Because the individual experiences the life-world differently from others, her model will also be different from the others' models of the world.

Schütz introduced a different notion of the life-world by drawing on Heidegger and assumed that *intersubjectivity* is always part of the life-world from the beginning on.

Basic principles on how the world works are shared amongst people. However, each person interprets 'actions' in the world subjectively and dependent on her experiences. This interpretation is based on the experiences of the individual as well as on the *relevance* of different aspects. This relevance depends on the motivation of an action, for example. Schütz distinguished between *because of* and *to* motivations. To understand the action of a fellow person, the individual tries to put herself in the other's position and imagines performing the same action. This view is of course based on the individual's experiences and the assumption that the fellow person sees the world similarly to her. Schütz assumed that there are common experiences everybody shares, others are related to social groups, and some experiences are purely individual. Common experiences therefore help to reach a common understanding.

We have referred to *the* life-world up to now. To be precise, Schütz differentiated between different 'worlds' in which we act. In his essay *On Multiple Realities* [Schütz, 1962], he introduced worlds where the *world of everyday life* (not to be confused with the life-world) is only one amongst for example a *world of dreams* or a *world of play*. Schütz also referred to these worlds as *provinces of meaning*. The sum of these worlds creates the life-world of an individual.

2.2.4 Theoretical Concepts: Creation of Meaning

> Sinn ist das durch Vorhabe, Vorsicht und Vorgriff strukturierte Woraufhin des Entwurfs, aus dem her etwas als etwas verständlich wird. [Heidegger, 1967, 151]

> Meaning is the 'upon-which' of a projection in terms of which something becomes intelligible as something; it gets its structure from a fore-having, a fore-sight, and a fore-conception. [Heidegger, 1962, 193]

Meaning is—according to Heidegger—always a part of our experiences in the world and depends on assumptions and experiences we made before.

In relation to our vignette, we discussed children and young people's meaning making processes. What we mean by *creation of meaning* is the process of the individual interpreting her actions depending on the context she acts in. For the girl Lena, who we met in our vignette, the context would be: her character and experiences, the mood she is in, her ideas, her mental model on how the robot works, her expectations towards the robots and many other factors. The interaction with technology as well as with other people is also a part of the context that influences how Lena creates meaning. Depending on these factors, she then interprets the reaction, e. g. the outcome of her program, and tries to make sense of it—that is the creation of meaning. In the world of play the children act in it is sometimes hard for the outsider to understand and follow the child's interpretation. The world of play is not necessarily related to socially constructed meaning of the physical world, as for example Langeveld pointed out.

> This form of meaning creation is not 'bound' to the physical world, and yet it is not structureless... Through play we see how the things in this world need not have fixed meanings. That which in the 'open sense-making' is a pencil now suddenly is a bridge, a road block, a soldier, or a house. [Langeveld, 1984]

Meaning making that involves an object, does not only involve the object, but the situation, the context the object is in at the moment, and also our intention, what we plan to do with it. Heidegger used the term *zeug*—often translated with *equipment*—to describe

> elements in the world turned into tools for our use. [Dourish, 2001b, 108]

Zeug or equipment

> is linked to other equipment in the way that it relies upon, works with, suggests, is similar or dissimilar to, or is otherwise related to other equipment. [Dourish, 2001b, 108]

Heidegger's notion of zeug extends tools that are work-related. Other objects can be zeug, too.

2.3 Towards Design: Embodied Interaction

Dourish, who we already referred to when discussing the social view on context, went a step further towards using a phenomenological perspective for the design of digital artifacts. He used the concept of *embodiment* as perspective to combine both: the technical and the social view on context. He then applied it to the design of computational artifacts. We will draw on the term context in this thesis, when referring to the general concept of the user and the ever-changing environment. We will introduce Dourish's concept of embodiment, because he applied the perspective to the actual design of computational artifacts. The design perspective gives us valuable insights for the design of learning environments in general. In his book *Where the Action Is* [Dourish, 2001b], Dourish laid out the foundation of his approach or framework and we will summarize some important aspects of his work.

Dourish defined important terms of embodied interaction and introduced the philosophical background he drew on—phenomenology. Then, he gave examples of domains where one can find embodied interaction. In the next step, he derived concepts and methods that can help the designer to design for embodied interaction, e. g. *familiarities* and *affordances*. We will introduce only those concepts that we consider important for this thesis. Dourish closed his book with a set of *design principles* (cf. [Dourish, 2001b, 162–187]—general guidelines a designer should keep in mind. We briefly summarize these steps.

Embodiment interaction for Dourish is composed out of embodied phenomena:

> Embodied phenomena are those that by their very nature occur in real time and real space. [Dourish, 2001b, 101]

He defined the term embodied interaction therefore as

> the creation, manipulation, and sharing of meaning through engaged interaction with artifacts. [Dourish, 2001b, 126]

The key idea of embodied interaction is: Actions take place in the world and cannot be separated from this connection. Therefore, it draws on phenomenology as theoretical background and applies Heidegger's view on creating meaning through action in the world. Meaning arises in an ad hoc situation out of the current context of the individual—her *lived experience* [Merleau-Ponty et al., 1968].

For us, this means that a learning environment for children and young people will always be used and understood differently depending on situation and intention of the user.

Dourish used the theoretical background to create a design framework that he then applied to two different trends in human computer interaction: *tangible computing* and *social computing*. For Dourish, both examples served the purpose to show how embodied interaction is implemented in the design of interactive systems. Tangible computing refers to the approach founded by Ishii and Ullmer [Ishii and Ullmer, 1997] which tries to move interaction with representations of data to the physical world. Because humans are familiar with acting in the physical world, Ishii and Ullmer assumed that tangible computing will make interaction with digital artifacts easier. Social computing, for Dourish, referred to the design process of interactive systems and should not be confused with social software, an application domain that we discuss later in section 5.1.

Dourish pointed out that both streams of current development might look different at first sight. When looking closer, they both share the idea of incorporating the real-world and the user's context into the design. In tangible computing, it is the fact that human beings are used acting on physical artifacts. In social computing, it is the consideration of real workplace settings and social practices in design, instead of relying on abstract descriptions on how a task should be done. We could also say, they both incorporate context into the design. For us, the important point of Dourish's argumentation is that both tangible and social computing are about the same thing: the user's context. We use this argument to show that in constructionist learning environments the material and physical side, but also the design process needs to be connected to the user's context. Furthermore, we would like to stress the point that embodiment as Dourish saw it is not necessarily physical, but applies in general. Dourish showed that not only with his examples but also with the following quotes:

> Embodiment is not about physical reality, but rather about availability for engagement. [Dourish, 2001a, 28]

> So, embodied interaction encompasses but extends beyond the physical, and so beyond the scope of physically-based ubiquitous computing or tangible interfaces. [Dourish, 2001a, 19]

Therefore, we can take Dourish's points and argue that connecting to the user's context cannot only be done through tangible or physical materials and the design process, but can also be extended to the design of software.

Dourish placed the hard part of the creation of meaning on the user's side—the user has to create meaning from the system's response rather than the system has to interpret the user's actions. For us, tagging and social software is therefore interesting, because the user creates more of the system's navigation herself. We will follow this approach throughout the thesis and ask how we can design to support the user in the process of meaning creation.

We would like to add another thought concerning the actors of an interaction. Dourish concentrated on the human-computer relationship as well, but he also pointed out that there is a user-designer relationship [Dourish, 2001b, 56] in every interaction with a computer. In his opinion, an interaction between user and computer is necessarily an interaction between user and designer at the same time. This is, because the designer's views and assumptions are always manifested in the system's design. This argument supports our hypothesis that children and young people should create their own environments and become designers themselves.

Another perspective on interaction can stress the point that interaction—especially social interaction—might as well take place between humans mediated through computation. Such an interaction takes place on different levels; there is always an interaction between human and computer and between user and designer, but the intention of the interaction might be a social interaction between humans in equal roles. We will call it a *human to human interaction*.

2.3.1 Design Concepts: Design Process

In his chapter on social computing, Dourish focused on methods that were developed to consider the real environment, e. g. a workplace setting of a user in the design of computer systems. The chapter is about an embodied interaction perspective to the design process, opposed to Dourish's chapter on tangible computing where he discussed the artifacts themselves. Observations in workplaces showed that often certain social practices evolve that support a task or change the way *how* a task is done. One of the examples Dourish introduced, was from air traffic control [Dourish, 2001b, 64–68]. The controllers work with cardboard strips that represent airplanes. The way they layout these cards on their tables represent the action of the airplane at the current point of time as well as the work flow: when to handle which plane. For the design of a system, these practices are very important. Imagine an implementation, where only the cardboard strips are digitalized without having the opportunity to change their position or order. Starting the design process by looking at the artifact (the cards) only, will most likely lead to such a result. Such a design would probably fail, because it did not consider the real practices, the practice on *how* people use it.

2.3.2 Design Concepts: Familiarities and Affordances

Dourish pointed out that interaction, for example in tangible computing, can be designed in two ways: The designer can design an entirely new physical artifact, or the designer can add computation or new functionalities to an existing one. A design of a new physical artifact requires a form that suggests how to use the device, an *affordance*. The concept of affordances goes back to the psychologist Gibson (cf. [Gibson, 1977]). Norman later on applied it to design of products and his notion of affordance is widely known in HCI and digital media. In his book *The Design of Everyday Things*, Norman defined the term affordance the following way:

> affordance refers to the perceived and actual properties of the thing, primarily those fundamental properties that determine just how the thing could possibly be used. [Norman, 2002, 9]

The suggestion to the user on how to use the artifact in a successful design is therefore implicit in the form of the products. It reveals itself to the user as an affordance.

Another way to design artifacts is to draw on familiarities. When an artifact is familiar to the user, she can make a connection to practices around the familiar product she already knows and transfer the practices to the new artifact. We will focus on a design that is based on familiarities. As we mentioned before, we assume that social software and textiles are good media for a constructionist learning environment, because they are familiar. The young users therefore have an idea how to use them and what to use them for. For example, especially for young people fashion and designing or combining clothes is familiar.

Familiarities and affordances do not only apply to the design of physical materials, but to software-design as well.

2.3.3 Design Concepts: Flexibility and Appropriation

In our vignette in section 1, we found the following situation: The Lego Mindstorms construction kit (see 3.8 for details) was designed with a certain usage in mind. The children and young people were supposed to build robots from the manual and program them with pre-defined programs depending on the model. However, the children and young people started to build their own models and wanted to use the pre-defined programs on what they just made. This did not work out, because we have a gap between the intended use and the actual practice around the construction kit. The children and young people tried to appropriate the technology, but in the case we described, they failed. The failure can have different reasons; we suspect that the design of the construction kit was not intended for flexible use.

The implications for the design of embodied interaction systems are a need for flexibility and the possibility to adapt to different users and environments as well as a familiarity [Dourish, 2001b, 99] to what we know from our world. Or of course, one would have to create such a good design that the affordances of the product can be perceived

by every user. This flexibility is necessary to fit into different contexts (in both respects—technical and social) of the user.

No matter how flexible a design is and what kind of usage the designer intended, the design of a digital artifact always includes abstraction and representation. In designing especially software, the abstraction is one of the key points. For a user, this means the abstractions of the designer have to be reinterpreted through coupling on her side.

> Embodied technologies are used to create and communicate meaning, managed at multiple levels through selective coupling; [Dourish, 2001b, 172]

In his design principles, Dourish put it the following way:

> Users, not designers, manage coupling. [Dourish, 2001b, 172]

Every individual has to go through the coupling process herself. This results in multiple interpretations and in different practices how a user will interact with the artifact. Looking at the process with the phenomenological background in mind, each user will interpret the artifact's reactions connected to her experiences resulting from her actions and also her motivations to use it. Common practices can evolve through common experiences, e. g. people working at the same workplace with the same tasks. Often these practices are very different from what the software designer imagined and planned. Such non-plannend practices can be called appropriations. Höök, for example, used the term to describe usage practices that evolve and were not intended by the designers [Höök, 2006]. We find them, for example, in our robotics vignette, where the children and young people try to use blocks for the Acrobot model for their own creation. We can find them also in tagging, where people use special characters, e. g. "#" as first characters of their tag to make it appear first in their alphabetically-ordered *tagclouds* (we will introduce tagging and tagclouds in more detail in 5.1.3).

2.3.4 Design Concepts: Dourish's Design Principles

We already mentioned the design principles that Dourish derived in section 2.3.3. From his theoretical foundation, he aimed at pointing out some principles that "observe or comment upon general features of embodied interaction that occur across a range of settings" [Dourish, 2001b, 162]. These six principles were called: "Computation is a medium"; "Meaning arises on multiple levels"; "Users, not designers, create and communicate meaning"; "Embodied technologies participate in the world they represent"; "and Embodied interaction turns action into meaning" [Dourish, 2001b, 162] (Dourish described each principle in more detail between page 163 and page 187).

2.4 Summary and Implications

After introducing what we mean by interaction and interaction design, we have explained concepts referred to with the term *context* and different understandings connected to the

term—the philosophical perspective. We have closer examined a human-centered perspective on the term which drew on philosophical underpinnings rooted in phenomenology. We have given a very brief introduction into phenomenology and some important concepts, such as life-world and intersubjectivity. While Husserl and Heidegger focused on the lived experience of the individual, Schütz worked on intersubjectivity—the question how to reach a shared understanding between several people through a shared life-world. However, we have not used a clear-cut definition of the term context, but wanted to show that we refer to a notion of context that depends on the current situation, activities and individual experiences.

Afterwards, we turned our attention to approaches that apply these underlying concepts to the design of digital artifacts—the design perspective. An outstanding example is Dourish's embodied interaction framework that we introduced. The framework—in line with the phenomenological approaches it is based on—puts the human and her world including the social practices she is really doing in the center. As a conclusion, we know that each user will interpret her own meaning depending on her context. To create a common meaning, two things are necessary: Shared experiences and shared basic assumptions on how the world works. As concrete consequences in the design of digital artifacts, we—as designers—have to (based on Dourish's design principles):

1. Research on the real-world setting of the user.

2. Integrate the user into the design process.

3. Draw on familiarities or affordances in our design.

4. Design flexible technology that fits into several contexts and allows multiple interpretations.

5. Count on appropriations that emerge. If possible, even design to support users in creating appropriations.

The way we understand it, the first three design guidelines address the problem of situating interaction in an ad hoc context that emerges. For example, in a workplace setting, several people experience the setting—they share the context. The last two items take the ad hoc creation of meaning of the individual into account and consider the changes in the context.

After laying out the basic assumptions of an interaction design perspective that draws on a wide notion of context with a phenomenological background, we will take a look at the learning theory we base our approach on. In the next chapter, the goal is to apply such a contextual perspective on constructionist learning and constructionist learning environments.

In our vignette (cf. section 1), we lined out a situation in which the three children wanted to program their artifact. The context is rather static on the one hand, the hardware and the model from the manual, they wanted to program with its sensors and actuators. On the other hand, the programming stands in relation with their idea of what the

artifact is supposed to do—dodge an obstacle in the example. For a successful programming, the hardware and the children's ideas how the dodging is supposed to happen, as well as their model on how programming works, need to come together. We will discuss the process in the next chapter. Furthermore, we found Lena being demotivated by the topic of robotics and her associations with robots—we see that the understanding of the concept "robotics" also is a part of the individual's creation of meaning in the current context. How constructionist theory and examples of learning environments designed with the theory in mind, try to make the object personally meaningful, is another question that we are set to answer in the next chapter.

Chapter 3

The Learning Perspective: Constructionism

While we discussed approaches that consider the design of interactive systems in general in the previous section, we are interested in their application to a special domain, namely learning environments. Learning environments are usually designed with an underlying learning theory in mind including certain assumptions on the human and the learning process (sometimes unconsciously). There are learning environments that rather follow a *behaviorists'* model and rely on question and answer repetitions on the one hand, or, on the other hand, *constructivist*-related learning environments that want the learner to actively shape and change their concepts on how the world works. In our vignette, the children and young people changed their concept on how they imagined programming (use the "dodge" block), because the experience with their robot showed that it did not work out.

We will rely to the different facets of constructionism as underlying learning theory in this thesis, because our vignette describes a learning scenario that makes sense in relation to constructionist theory. To introduce our understanding of constructionism, we start by discussing basic concepts and variations of the learning theory and then take a closer look on learning environments that were developed to apply the theory—the design of constructionist learning environments. Our foci are creation of subjective meaning, as well as the creation of a common understanding that is why we consider extensions of constructionism towards social learning.

The main theories related to constructionism are Piaget's constructivism, Papert's constructionism and Vygotsky's *social constructionism* (also called *social development theory*). Piaget's constructivism is the model Papert uses as basis. Piaget's main contribution[1] Papert builds on, is the idea that children and young people actively construct their own knowledge and only change their concepts on how the world works when these concepts do not work anymore (cf. [Ackerman, 2002]). The implication is that to create a successful constructivist learning environment, we cannot lecture children and young

[1] We refer to Piaget's contributions relevant to Papert. Many people know Piaget, for example, because of his *developmental stages theory*.

people what is right or wrong, but create situations where the children and young people's concepts are challenged and stop working. Then the children and young people eventually replace their old concepts with newer, hopefully more appropriate concepts. This process of *accommodation* is a complicated one, in which children and young people might go through confused phases.

3.1 Theoretical Concepts: Papert's Constructionism

Papert is a mathematician from training who became professor for artificial intelligence at the MIT (Massachusetts Institute of Technology) in the nineteen sixties. Inspired by his work with Piaget, he founded the Epistemology and Learning Research Group[2] at the MIT's Media Lab and started to use computers for education as a pioneer. His first works on learning were published in the nineteen seventies. Papert's starting point was learning and teaching of mathematics and the role of the computer for him was to make abstract mathematical concepts more concrete. He referred to the metaphor of gears to illustrate the idea. Papert [Papert, 2000] used gears in his imagination as *objects-to-think-with* [Papert, 2000] that helped him to reflect upon the underlying mathematical concepts.

> Gears, serving as models, carried many otherwise abstract ideas into my head. I clearly remember two examples from school math. I saw multiplication tables as gears... [Papert, 1980, vii]

Learning by designing is the key idea of Papert's theory of constructionism. His approach is related to and based on Piaget's constructivism in the following sense: Both theories understand knowledge as something that people actively construct by interaction with objects or people. In his theory, Papert added the idea that it supports the learning process when ideas are externalized in objects and these objects people interact with are personally meaningful and sharable (objects that allow communicating with other people about their ideas). The following quote makes the difference between both approaches clear:

> Constructio*n*ism—the N word as opposed to the V word—shares Constructi*v*ism's view of learning as building knowledge structures through progressive internalization of actions... It then adds the idea that this happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe. [Papert and Harel, 1991, 1]

We would like to stress the point that the object is not necessarily of physical nature, as Papert makes clear in the quote above. The object can also be a theory not a graspable object. The Logo language and the well-known *turtle graphics* are examples of such non-physical objects.

[2] http://el.media.mit.edu/

> Constructionist theory goes beyond Piaget's constructivism in its emphasis on artifacts, asserting that meaning-construction happens particularly well when learners are engaged in building external and sharable artifacts. Thus, there is a convergence in the fields of design and learning, with a natural intersection in the study of learning-through-design. [Kafai and Resnick, 1996, 4]

The object has to be an external entity, because externalization plays a great role in constructionist learning among other things to communicate one's ideas to others. Externalization and internalization in constructionism are interrelated in the following way:

> ...when the learner is engaged in the construction of something external or at least shareable... a sand castle, a machine, a computer program, a book. This leads us to a model using a cycle of internalization of what is outside, then externalization of what is inside and so on. [Papert and Harel, 1991, 3]

According to this quote the learner, who is at the same time the designer, would then always internalize knowledge from what she learned during the construction and externalize this knowledge in the ongoing design process around the object. For the learning process, it is necessary that the learner steps back from the actual creation in the cycle of internalization and externalization. Ackerman called the learning process through *stepping-out* before *diving-in* again:

> Cognitive growth as an ongoing dance: diving-in and stepping-out. [Ackerman, 1996, 28]

This construction of knowledge and meaning is the key to successful learning in constructionism and the role of the design process is the creation of subjective meanings. We set construction of meaning on a level with meaning creation that we discussed in more detail in section 2.2.4. The process of meaning creation is similar to a bricolage. We apply the term bricolage in our thesis in a notion from Lévi-Strauss [Lévi-Strauss, 1968], who used it to describe the process of taking objects and functions out of their original context and put them into a new context. In connection to smart textiles, we use the concept again in section 5.2 to describe the design process. The designer takes functions of an existing object and tries to understand their meaning. She afterwards recontextualizes the understood meaning in making a new object. In the domain of textiles and fashion, this can for example be an LED that is worn as a piece of jewelry in a necklace. Usually LEDs are connected to the world of electronics and not fashion. The designer of the necklace made a bricolage by connecting the previously unconnected items LED and necklace to a new kind of jewelry.

> Both design theorists and learning theorists now view construction of meaning as a core process. In this view, design involves building a relationship between designer and object.
>
> Designers sort out what objects mean to them or others, and they selectively connect features of an object and features of a context into a coherent unity. The relationships that designers build with objects or situations constitute the

> core focus of design theory; the final artifact itself is secondary. Design is now viewed as the process through which a designer comes to understand not only objective constraints but also subjective meanings. [Kafai and Resnick, 1996, 4]

The term bricolage is also used by Turkle and Papert (cf. [Turkle and Papert, 1991]) in connection to constructionism to differentiate styles of learning. They identified two types of learners and programmers: the *planner* that thinks the whole way of creating (for example of a program) through in advance and the *bricoleur* who changes the model as she goes along.

The focus of constructionism (or the early writings in constructionism) is on the individual, who creates subjective meaning by a cycle of internalization and externalization, always manifested in an object. Because children and young people typically decide on their own project "constructionist learning environments encourage multiple learning styles and multiple representations of knowledge" [Kafai and Resnick, 1996, 3]. Each child can pick up topics and materials important to her. From a contextual point of view, the learner builds the artifact and always interprets results of the construction or programming depending on her actual context. Furthermore, although the object-to-think-with can as well be a virtual artifact, creating physical artifacts—for example Lena's robot in our vignette—can be part of the constructionist learning environment. Sharing Dourish view, we consider the physicality or tangibility as a good way to bring the learning object in the learner's context. We notice that in the basic writings of constructionist theory the sharing of context or the creation of meaning in a real life context is not an issue which is discussed a lot. In the very idea to create an object that is external and therefore shareable, a shared creation of meaning is implicitly assumed, though.

Coming back to our vignette, we can see that the constructionist learning theory is used in the learning setting. The children and young people actively create their own object-to-think-with; they build and program the artifact. The concept of the vignette fails in addressing the personally meaningful part—Lena would prefer to build something else.

These are the basic assumptions and concepts of constructionism. We now start to discuss novel ideas and extensions of constructionism that aim at the creation of social learning experiences and focus on sharing of meaning in communities and discuss the perspective on sharing explicitly.

3.2 Theoretical Concepts: Papert's Samba Schools and Beyond

Although interaction with other people and the environment was not in the focus of Papert's early writings through the importance of externalization constructionism always considered interaction. Papert saw programming as a rather single activity, but wanted to encourage communication, community learning and collaboration from the beginning on [Papert, 1980, 219]. The interaction for Papert had different objectives: the sheer

fun of sociability as well as the possibility to ask for help in a common language that is understood within the community (Papert referred to the Logo language with the "common language"). His model of a successful community based learning environment was a *samba school*, as we can find it in Brazil. People from different backgrounds and ages come together, because they share the same interest and meet to perform a certain activity together. The model referred to real-world communities, not to virtual ones, at the time Papert mentioned it first.

These samba schools are often referred to by researchers, who extend constructionism with a social component, as indicators for Papert's interest in community learning (cf. [Bruckman, 1998] or [Shaw, 1996]). Papert used the term school although what he described as a samba school is rather comparable to informal learning communities in western countries, sports clubs for example. People of different ages and abilities come together in an informal setting.

> From this point of view a very remarkable aspect of the samba school is the presence in one place of people engaged in a common activity—dancing—at all levels of competence from beginning children and young people who seem scarcely yet able to talk, to superstars who would not be put to shame by the soloists of dance companies anywhere in the world. The fact of being together would in itself be "educational" for the beginners; but what is more deeply so is the degree of interaction between dancers of different levels of competence. From time to time a dancer will gather a group of others to work together on some technical aspect; the life of the group might be ten minutes or half an hour, its average age five or twenty five, its mode of operation might be highly didactic or more simply a chance to interact with a more advanced dancer. The details are not important: what counts is the weaving of education into the larger, richer cultural-social experience of the samba school. [Papert, 1976]

The samba schools put learning into a larger context—the cultural-social experience as Papert described it. Learning is not isolated from the cultural setting around the activity. Deploying our contextual perspective, the samba school is situated in the real-world, the learning experience is interpreted and makes sense for the learner in this current context—also for the other people who share the situation and the context.

Coming from the samba schools in the real-world, Papert and his colleagues [Papert and Resnick, 1995] realized the new possibilities the Internet offered for constructionist learning environments and interaction between learners. They envisioned a Constructopedia. The Constructopedia, as they called it in a research proposal [Papert and Resnick, 1995], was meant to be an Internet database that allows children and young people as designers to share their projects and ideas. It was a very new and innovative idea around that time, the Internet was not very common and wikis[3] or Internet

[3]Wikis are software tools that allow users to create and edit web pages in an easy way without having to know complicated programming or markup languages. A community usually uses wikis to add content and to create a knowledge database. The most popular example is Wikipedia (http://www.wikipedia.org) that has the value of an encyclopedia for many people nowadays.

forums were not used by everybody yet. Fred Martin—one of the researchers around the Programmable Bricks that we will discuss in section 3.8.3—related the first discussion of the Constructiopedia to the time when Hypercard[4]—a hypermedia system by Apple that was very popular before the WWW—celebrated huge success [Martin, 2006]. The idea of the Constructiopedia is quite similar to a wiki like the Wikipedia.

The Constructopedia[5] was meant to be a shared knowledge base. Children and young people would use it for help and support. The research proposal itself was not granted, though, but its idea was taken up and implemented in different projects and student's theses e. g. Pearls of Wisdom [Chapman, 2003] or Shook's Master thesis [Shook, 2000].

The aspect of support and help by sharing contents on the Internet or generally speaking on computer networks was taken up by Resnick, who developed a theoretical perspective called *distributed constructionism* that put sharing and discussions on creations in the focus; we discuss the approach in more detail in section 3.6.

Shortly after Papert and Resnick wrote the Constructopedia proposal—in 1996—the book *Constructionism in Practice* [Kafai and Resnick, 1996] was published and added the social learning or community learning aspect to Papert's constructionism or—better say—made the social aspects we already found in the samba schools in constructionism explicit. The book had a chapter on *Learning in Communities* that discussed several projects that focused on social learning. MediaMOO[6] and MOOSECrossing[7] from Bruckman and Resnick were presented as examples of a virtual communities [Bruckman and Resnick, 1996], while Shaw [Shaw, 1996] emphasized building real-world communities in an inner city urban community. Shaw's understanding of social constructionism extends community learning and we discuss it in more detail in section 3.4.

Evard [Evard, 1996] argued that sharing of creations through a real-world community with an additional virtual space leads to a deeper understanding of other persons' ideas and discussed her analysis of question and answers children and young people asked in a virtual discussion space accompanying a project, where children and young people made their own video games.

For us, the articles about new forms of community learning are important, because they show a direction from Papert's samba school—a constructionism centered on community learning: virtual as well as real-world communities that build around constructions not only for support; also for fun and feeling as a social being.

We will shortly look at these additions or extensions to Papert's constructionism from the contextual point of view. Before doing so, we would like to give a brief introduction into another approach related to constructionism, namely Vygotsky's social constructionism.

[4] http://www.apple.com/hypercard

[5] Interestingly the term is quite well-known, because Lego took it up as name for their construction manual that was part of the Lego Mindstorms 1.5 construction kit.

[6] http://www.cc.gatech.edu/ asb/MediaMoo/

[7] http://www.cc.gatech.edu/elc/moose-crossing/

3.3 Theoretical Concepts: Vygotsky's Social Constructionism

We just discussed how social and community aspects became more important and more explicit in Papert's constructionism in a development process from Papert's first writings. Vygotsky, on the other hand, made the social and interaction explicit aspects central to his theory from the very beginning. Vygotsky lived and worked before Papert's time and published most of his work in the nineteen twenties and thirties.

Vygotsky is not necessarily a constructivist, but his work in developmental psychology added a new perspective to constructivism (cf. [Boudourides, 2003]). Vygotsky emphasized the role of social interaction in the learning and development process. He shared the view of Papert and Piaget that the learner actively constructs her knowledge out of experiences in the world, but his focus was rather on the interaction with other people (adults as well as peers). The child Vygotsky imagined, builds a concept of the world from what it sees and experiences in social interactions. A special role for Vygotsky played language. One of the concepts widely known about Vygotsky is his *zone of proximal development* (ZPD) that he defined as:

> ...the distance between the actual developmental level as determined by independent problem solving and the level of potential development as determined through problem solving under adult guidance, or in collaboration with more capable peers [Vygotsky, 1978, 86]

Typically, through interaction the individual can develop further than without—there is a ZDP that depends on the individual child. This is why Vygotsky placed emphasis on encouraging interaction of the learner to both peers and adults. His approach is a common basis for current (learning) theories that focus on the differences Vygotsky made in his zone of proximal development. An example is *cognitive apprenticeship* [Collins et al., 1989], where a *master* guides a learner through the learning process. The master reduces the teaching step by step towards giving more responsibility to the learner. This is only one possible way of interpreting Vygotsky's ZDP, an interpretation also called *scaffolding*.

Other approaches are based on Vygotsky too, but rather on situating the learning process in a context, for example *situated learning* [Lave and Wenger, 1991] or *communities of practice* [Lave and Wenger, 1991]. We already mentioned an approach of *situating* when referring to Suchman (in our section 2.2.1 on context), and Lave and Wenger's concept is similar. Lave and Wenger [Lave and Wenger, 1996, 144] mentioned two other possibilities of interpreting Vygotsky's ZDP—the *cultural* and the *collectivist* interpretation. According to the authors, the cultural interpretation understands the ZDP as the distance between everyday knowledge and scientific knowledge—Vygotsky differentiated between both. The collectivist notion of the ZPD sees the zone as distance between everyday experience of the individual and the interpretation of actions in society. Situating learning into the context of the real-world and into communities that share a practice creates a common context of the members the community has. We mention the approach

here, because it is a good example, how to relate to the social context in a learning theory and we aim at doing so similarly, but with a greater focus on the object. Also, in our section on social software we attempt at looking closer on such a community of practice and especially on the practice of tagging.

3.4 Theoretical Concepts: Social Constructionism Inspired by Papert and Vygotsky

Vygotsky-inspired concepts fit well with our understanding of context, they are in addition to the emphasis Papert placed on the object our theoretical basis for the learning perspective. The combination of both Papert and Vygotsky's approach with the emphasis on externalization through the object can be the basis for a contextual perspective.

Some researchers worked on exactly this—trying to bring together Papert's focus on the object and the Vygotsky-inspired constructionist learning communities. This is, of course, an interesting perspective for us. When comparing Papert's samba schools and the Vygotsky inspired communities of practice, we can see that there are assumptions both approaches share. One can also understand a samba school as a community of practice involved in a construction activity. For this thesis, we will especially refer to Shaw, who drew on Vygotsky in a constructionist project. Shaw comes from a constructionist background and is influenced by Papert—as well as by constructionist applications in the context of the MIT. Shaw [Shaw, 1996] (in line with other researchers, see the previous section on constructionism in our chapter 3) interprets constructionism as an addition to constructivism, as an explicit perspective that considers making and sharing meaning through the design and creation of objects. On the other hand, he proposed a wider notion of social constructionism taking Vygotsky's theory into account.

Social constructionism in his point of view is constructionism that highlights the role of objects as socially constructed which they are through the (social) setting they are created in. For example, materials, the basis for any construction work, have a meaning and a function in society and also in a certain community that will shape and influence the design process. The materials are *cultural materials* [Shaw, 1996] related to certain practices and ideas. To say it in our words—through the design process and the bricolage within the design these relations change and new relations are created. Shaw tried to create social settings—a community—in which the social creations are constructed similar to the constructionist culture which we discuss in the next section. He differentiated between certain types of social constructions namely [Shaw, 1996, 181]:

1. *Social relationships:* Shaw referred to the different kinds of social relations in a community: friendships, families and all the different variations:

2. *Social events:* According to Shaw, these are events where everybody in the neighborhood can take part, for example a block party.

3. *Shared physical artifacts:* A shared physical artifact belongs to the whole community and the community keeps an effort to maintain it, e. g. a public basketball loop.

4. *Shared goals and projects:* Typically, these shared goals and projects help the whole community e. g. keeping the neighborhood clean.

5. *Shared cultural norms and traditions:* Shared practices arise in a neighborhood; dialects, language or a clothing style are examples.

Creation of these constructions takes place within a community, in Shaw's case a neighborhood and through this real-world community a shared meaning of the artifacts arises and the social constructions are build. Shaw pointed out that this social setting is something dynamic and the relations are changing all the time as we saw in our section 2.2.1 on context. Through the constructions Shaw referred to, the neighborhood builds up its shared context. Shaw uses his notion of social constructionism to place construction activities in a neighborhood community.

3.5 Theoretical Concepts: Constructionist Culture

Bruckman studied the MOOs (text-based adventures, see section 3.8.5 for details) MOOSE Crossing and the MediaMOO in her dissertation [Bruckman, 1997]. For her, social interactions are an important element of constructionist learning, because they can help to provide a surrounding in which learning can take place [Bruckman, 1997, 17]. She argued that community building should be supported to provide a positive context to learn, referring to Papert's samba schools. As results of her studies, she stressed the inseparability between the learning process and social activities [Bruckman, 1997] and pointed out that:

> Social context is of central importance to any learning experience. One of the strengths of networked learning environments is their ability to help integrate a supportive social context with the computational context. [Bruckman, 1997, 117]

The learner is of course always in a social context, even when being alone in front of a computer, but in Bruckman's case she referred to the social context of *constructionist cultures*. Bruckman used the term constructionist culture for communities where people come together to make things.

> A particularly felicitous type of community often emerges when people are brought together to construct things. Samba schools in Brazil are an excellent example. Creating a presentation for Carnival brings together a group of people who might not otherwise meet.... This chapter argues that communities in which people are making things often take on a special quality. Such communities have what I will call a constructionist culture. [Bruckman, 1997, 159]

The constructionist culture is a concept related to the samba schools, but extends the concept, because Bruckman combined the physical setting of a samba school with the mixed reality of a virtual community. The social experiences Bruckman focused on, were fun and joy of sociability.

3.6 Theoretical Concepts: Distributed Constructionism

Around the same time Bruckman published her doctoral thesis, Resnick introduced the concept of *distributed constructionism* where he added three main activities to Papert's constructionism [Resnick, 1996]. These activities can take place virtually in a computer network and are supposed to help knowledge building through social interaction. As activities Resnick defined:

1. *Discussing Constructions:* Resnick suggested that children and young people can use the Internet to discuss what they created and referred to "electronic mail, newsgroups, and bulletin boards" [Resnick, 1996] as suitable media for the discussion.

2. *Sharing Constructions:* For Resnick, sharing in the first place meant that the creators could publish their creations in some kind of representation on the Internet. During the time of his writing, it was not possible to share interactive creations, now the situation changed: For example with Scratch (see section 3.8.4 for details) children and young people can not only publish a photo or video of their game, but the whole interactive object.

3. *Collaborating on Constructions:* Collaborating (on constructions) is the highest level of distributed constructionism, according to Resnick. It refers to situations where children and young people create things together in some kind of virtual space. As an example, Resnick referred to MOOSE Crossing that we will discuss in more detail in section 3.8.5.

Distributed constructionism focused on building virtual communities for knowledge sharing.

3.7 Theoretical Concepts: Summary

We introduced the key ideas of constructionism, namely that the individual learner creates meaning through designing—and actually creating—an artifact by assembling and re-assembling meanings. A cycle of externalization and internalization takes place within the design and learning process. The learning happens during and through the design process. This assumption is common ground in constructionism and is based on constructivism by assuming the learner actively creates her own model on how things work through their own experience.

The perspective on interaction between the learners or between learner and e. g. teacher came into the attention of writings in constructionism as the theory developed. Papert already pointed into the direction by discussing the samba schools and his successors made the importance of interaction and sharing more explicit through several new perspectives that focused on the social aspect in the ongoing development of the constructionist theory. We are interested in the extensions towards social interaction, because the creation of a common understanding—intersubjectivety as Schütz called it—is the basis for having a shared context.

The constructionist extensions which we introduced, focused on knowledge sharing and exchanging objects through the Internet as well as on community building for fun and sociability. The approaches aimed at situating the learning process into a community or a cultural context in which the construction activity takes place.

The different directions did not explicitly discuss the process of creation of meaning within the community. They seem to assume that a shared understanding is created through the objects that are created in the shared space or—in the rather community oriented perspective—through the interaction that takes place between people in the real as well as virtual community setting.

The extensions of constructionism and current learning theories based on Vygotsky have some similarities that we discussed in connection to Vygotsky's basic assumptions. They try to situate the learning process into a context and into a social setting. We would like to point out Shaw's approach separately, because he applied a slightly different perspective that is also inspired by Vygotsky. Although the basic assumptions are the same as in the community perspective—learning improves through social interaction—he extended the other two approaches: In his social constructionism, he focused on non-human actors e.g. materials as large part of a community setting. He also saw relations between actors as a constantly changing dynamic situation. Meaning arose from an emergent set of relations—the ad hoc situation as Suchman called it (see our section 2.2.1 on context for details). Within a community shared traditions and norms exist that have an impact on how members of the community create meaning for themselves. Shaw's social constructionism fits well with our view on context (that we introduced in section 2.2.1). For our learning environment this means:

1. Software as well as physical materials are part of the learner's context and their meaning depends on the individual and the context.

2. We do not know in advance, how children and young people understand and use material and software, so we have to design technology that they can adapt to their context and that considers a changing context and interaction as important from the basic design on.

After having introduced basic assumptions of constructionism as a theory, and extensions towards a contextual and social perspective, we will start to introduce examples of constructionist learning environments, especially programming environments. As in chapter 2 on context, we apply both: a perspective on theory (what we just did) and a perspective on the actual design of technology.

How are the environments designed on a conceptual level to implement constructionism is our main question that we have in mind when exploring and introducing examples of constructionist learning environments.

We will give a very brief overview over the development and history of constructionist programming environments. We will then discuss some example environments in more detail under a perspective on context and creation of meaning in a way we introduced the term in section 2.2.1. Our focus in this section is on how the designers of the learning

environments tried to ease meaning creation for the child or young learner that interacts with the system.

3.8 Design Concepts: Programming in Constructionist Learning Environments

Although a constructionist environment not only consists of technology, but of the whole setting (including material, pedagogical concept, teachers and so on) that has a great impact on how children and young people use and understand the tools, we will focus on technology—especially on programming environments—and its development in this section. Because some technologies are linked to certain contexts or communities, we will discuss those contexts or communities as well. However, we won't go deeper into pedagogical concepts relating to those technologies (for example, on how to introduce the technologies or how to support children and young people in finding their personally meaningful project ideas) at this place. We do consider these aspects as important, though.

3.8.1 Design Concepts: Playful Learning and Hard Fun

Most constructionist learning environments focus on engaged *playful learning*. We use the term playful learning in a notion by Resnick, who contrasted playful learning and *edutainment*. From his point of view, the disadvantages of edutainment were:

> When people think about 'education' and 'entertainment,' they tend to think of them as services that someone else provides for you... That is a distorted view. In fact, you are likely to learn the most, and enjoy the most, if you are engaged as an active participant, not a passive recipient. [Resnick, 2004]

Instead of giving a definition of the term playful learning, Resnick gave an example of a girl that was not too interested at school, but got engaged in building her own technology project in an after school activity. She built a marble machine, constantly changing and testing the design and, finally, won prizes at the school's science fair [Resnick, 2004]. Papert coined the term *hard fun* for this kind of experience. He argued that fun can arise from a challenge.

> A teacher heard one child using these words to describe the computer work: 'It's fun. It's hard. It's Logo.' I have no doubt that this kid called the work fun because it was hard rather than in spite of being hard. [Papert, 2002]

Fun is seen as crucial component for many of the constructionist learning environments and through the playful learning children and young people can be empowered and motivated to learn. To create environments that support such a learning process Bruckman et al. gave recommendations for environments for children and young people. The following quote stresses the point that children and young people like

> to be in control and not to be controlled, to create things and to express themselves, and to be social and to collaborate. [Bruckman et al., 2001, 5]

3.8.2 Programming and the Role of Programming

Constructionist learning environments include very different activities that have in common that children and young people create something, some external object-to-think-with by themselves. Children and young people create and program physical objects, for example. They might also create programs in purely virtual environments, but also create videos and other media or games. Common kinds of constructionist learning environments are so-called *construction kits* that Resnick and Silverman described in a very broad sense as:

> ...systems that engage kids in designing and creating things, sometimes on the screen, sometimes in the physical world, sometimes both. [Resnick and Silverman, 2005, 117]

We would like to focus on construction kits that combine physical materials with a programming language and that are made to allow playful learning. In chapter 1, we pointed out the special sensation of programming tangibles materials. In most cases programming takes place in the virtual world with a screen-based programming language. Novel applications in some cases even place programming in the physical domain and we will discuss them briefly later on. The relation between physical artifact and programming environment in the screen-based variation is particularly interesting for us. We mentioned the problems that we encountered while working with such construction kits before, in our central problems summarized in the basic vignette. The children in our scenario misinterpreted the representation in the software and ended up frustrated, because they did not understand what they did wrong.

As soon as children and young people write programs that run on physical artifacts, programming and physical artifact are linked and hard to separate and, as we already mentioned, the physical object becomes part of the context the child has to understand in the programming environment. The characteristics of the physical object are partly defined through its behavior—the programming. And the physical object serves as runtime environment and debugging engine for the programs the children and young people write.

Constructionist learning environments share common activities in the sense that they all aim at enabling the children and young people to build something themselves to gain an understanding of a more abstract concept. Programming is an activity central to constructionism, because Papert's first application was the programming language Logo that he invented to ease the learning of mathematics. However, Papert's starting point of trying to make math more concrete, is not the only goal of constructionist learning environments, especially programming, anymore. Understanding of mathematical, but also technological concepts, e.g. the flow of current, or underlying principles on how a computer works—the algorithm—, can be goals of making children and young people

learn programming. Schelhowe considered programming (or the construction activity) as access to these principles:

> Es geht weniger darum, die Grenzen ziehen zu können, sondern darum zu begreifen, welche Prinzipien hinter der Herstellung der smarten Interfaces liegen. In den (eigenen) Konstruktionen werden die eigenen Imaginationen und die anderer, die hinter diesen Interfaces liegen, sichtbar und kommunizierbar, damit auch kritisierbar und korrigierbar. [Schelhowe, 2007, 152]

> It is rather about understanding which principles underlay the creation of smart interfaces than to be able to set boundaries. Through the construction the individual's as well as the others' imaginations that underlay the interfaces become visible and external, through that also criticizable and corrigible. [Schelhowe, 2007, 152]—translation

Tholander and Fernaeus stated that:

> There are two basic motivations for designing programming tools for children, to make children smarter and to empower children to create interactive simulations, toys, and games. [Tholander and Fernaeus, 2004]

We do not totally agree with Tholander and Fernaeus. "To make children smarter" can be a motivation when making math more concrete, but we see allowing different ways to understand complex concepts as a third motivation. Still, the statement shows a change in the role of programming. The idea of empowerment to create digital media is a new perspective on programming compared to Papert's vision. In many constructionist learning environments we see a mix between the two motivations. The role of programming in education as well as the reason why children and young people should learn to program is heavily discussed.

Classifications of Programming Environments

To learn programming is considered as something that is hard to learn, because the learner has to go through an abstraction process. There are several approaches and examples to design programming environments to ease the process. Although not all approaches stand in relation to constructionism, they all engage the learner into a construction activity, namely programming.

According to Kelleher and Pausch [Kelleher and Pausch, 2005], programming can be taught to teach programming itself, as a tool to understand other complex topics (in Papert's example it was mathematics, but it could also be a complex reaction in chemistry) or to be able to configure certain hard- and software as an end-user[8]. Depending on the motivation of teaching and learning programming the goals are also different. When a future software developer learns programming, correct models and good programming style might be more important than when a child learns to make her own video game in

[8]End-user programming to configure devices might become a very interesting topic in relation to smart textiles.

an approach that focuses on empowerment and on enabling children and young people to create media.

There is a number of programming languages especially made for educational purposes with different learning theories and approaches in mind. There are general-purpose languages for beginners as well as special-purpose languages for certain application domains. In this thesis, we will refer to these special-purpose languages as *domain specific languages* (DSL)[9]. We see a language to program a special hardware platform, such as Handy Crickets[10], as a DSL, too.

From the large range of different motivations to teach programming and learning goals, not only the number of programming languages is huge, there also very different ways to order and classify the languages.

Kelleher and Pausch [Kelleher and Pausch, 2005] developed a taxonomy for programming languages for novices classifying the languages models by how the user expresses programs. They came up with "Simply entering code" (simplified languages, prevention of syntax errors) and "alternatives to typing programs" (using graphical or physical objects to program, create programs using interface actions, provide multiple methods for creating programs) (cf. [Kelleher and Pausch, 2005]). Their taxonomy is very complex; the expression mode is only one level of classification.

Guzdial [Guzdial, 2004] offered a different taxonomy driven by the history and development of programming languages for novices. He differentiated between the Logo family, a "rule-based" family and "traditional programming-language" family that he defined as:

> ...novice programming environments, which tried not to change the language, but instead provide new student-centered supports for existing programming languages. [Guzdial, 2004, 128]

An example of such a language is Pascal[11].

Results and Further Approach

We will use a classification similar to Guzdial, focusing on the first family and a direct connection to constructionist learning theory and look at examples from three families in detail.

1. The biggest family we take a look at is the Logo family—the very first example of the design of constructionist learning and programming environments—and we especially focus on Logo languages in connection to physical artifacts.

2. Then we take a closer look on Smalltalk successors with the novel Scratch language. The Smalltalk family is strongly inspired by LOGO and Kay as main developer relates to Paper's constructionism which is why we discuss the Smalltalk family.

[9] A DSL is a programming language designed to work on specific tasks. An example could be a specific script language for an office application

[10] http://www.gleasonresearch.com

[11] http://www.moorecad.com/standardpascal/

3. As third example, we chose MUDs and MOOs that are also inspired by Logo, but developed into a direction that focuses on community building. MUDs and MOOs stand in direct relationship to constructionism.

In this thesis we apply the following view: Children and young people should learn about circuits and programming to gain a basic understanding of how technology works. In our view, this understanding can empower them to take part in the participatory digital culture as active creators, instead of only as passive recipients. We see the learning process as successful, when children and young people broaden their field of possibilities, when they perceive they have new possibilities to create and change their world. In chapter 7 on evaluation, we will discuss some examples. Children and young people might discover that programming is something *they can do* and maybe even consider working in technology-related field as new possibilities.

3.8.3 Design Example: The Logo Family

Logo can be seen as a starting point for the development of educational languages as well as the starting point for constructionist learning environments, what is not necessarily the same, as we pointed out before.

The Logo language itself is written in LISP. Especially the Logo dialects for the programming of physical artifacts are rather imperative. There are many different Logo dialects. The idea behind Logo is to offer a simple set of commands to create an (visual or physical) output. The best known visual output from Logo is the *turtle graphics*—a turtle that could move controlled by commands such as:

Listing 3.1: Creating a square with Logo

```
1 forward 50
2 left 90
3 forward 50
4 left 90
5 forward 50
6 left 90
7 forward 50
8 left 90
```

Figure 3.1: Output of the turtle

The turtle controlled by the program left a drawing on screen, according to its movements, as you see in the very easy example of a square in figure 3.1.

Design Example: Physical Platforms and Their Programming

The Logo development is strongly connected to the development of physical platforms. We see smart textiles as a physical platform for programming as well. That is why we introduce the related platforms here. Papert originally invented the *floor turtle* as output device for Logo. Logo commands were entered on the screen and transferred to the floor turtle, a physical turtle connected to a computer with a cable. The floor turtle had a pen attached that moved across the floor and drew on the floor according to its programming while moving. The floor turtle was too expensive to be used at school which is why the turtle graphics are better known.

With the Programmable Logo Brick and the Red Brick (cf. [Martin et al., 2000]), Martin and his colleagues developed the first successful programmable smart physical object in the educational domain that was available for wider use. The novel approach was its ability to operate autonomously, without being physically attached to a computer by a cable. We will use the term *brick* to describe different kinds of programmable physical objects. The bricks include a micro-controller that can run the children and young people's program as well as inputs and outputs for sensors (e. g. simple electronic switches) and actuators (mostly motors that allow building moving vehicles). Brick Logo was the first Logo language to program such tangible artifacts. The successor Lego TS Logo worked with the Programmable Logo Brick and the Red Brick as well as with Lego Mindstorms.

The graphical user interface to the programming language was very reduced and consisted of three elements: the monitor to receive values from the brick (e. g. values of an attached sensor), a shell to type in simple commands and send to the brick and a space to write complex programs in (cf. [Martin et al., 2000]). Such a "complex" program (in the Cricket Logo DSL) could be:

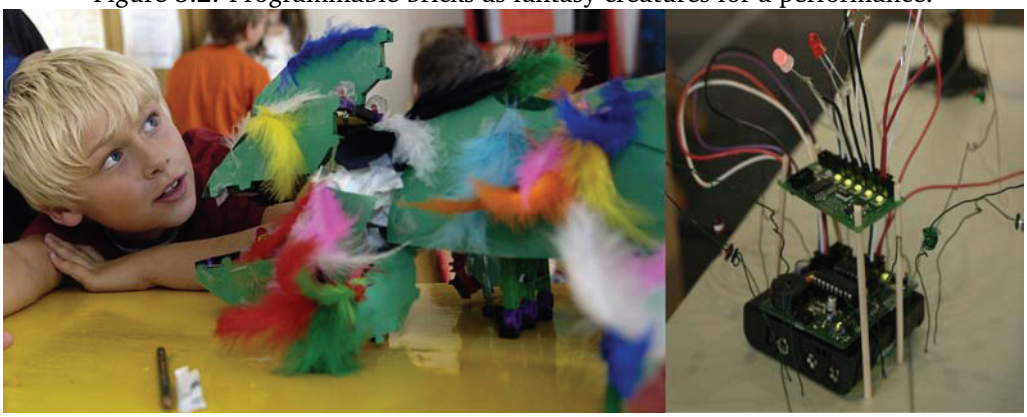
Listing 3.2: Easy Cricket Logo program

```
1 to test
2 a, onfor 10
3 beep
4 b, onfor 10
5 end
```

The program would switch on an actuator (let's say a motor on the left side of a vehicle) for 10 milliseconds, beep once, then switch on another actuator connected to the pin labeled "b" for 10 milliseconds.

The Programmable Logo Brick (and the commercial variation Lego Mindstorms that followed later) by Resnick et al. was initially designed to support very different activities around ubiquitous computing [Resnick et al., 1996]. As a reference to Papert's and Solomon's Memo *Twenty Things to Do with a Computer* [Papert and Solomon, 1971], Resnick published an article called *Twenty Things to Do with a Programmable Brick* [Resnick, 1993]. Example applications included using the brick as a musical instrument or controlling some parts of the children and young people's house with it, e. g. by tracking the number of people that enter a room through a light barrier.

Figure 3.2: Programmable bricks as fantasy creatures for a performance.



The commercial version Lego Mindstorms was rarely presented as a kit for various applications in ubiquitous computing. To prevent misunderstandings, we would like to add that it is of course possible to build very different applications with Lego Mindstorms, we only argue that the design of the kit focused on a use for robotics[12]. Lego Mindstorms were rather sold as a product for robotics, as for example the name of the corresponding official programming language, Robotic Invention System (RIS), implies. The hardware design of Lego Mindstorms and the Red Brick was also changed and adapted for the needs of the robotics domain. The number of inputs and outputs, for example, was reduced to four, but an interface to control motors was added. Due to the circuit design of the parts for Lego Mindstorms no standard electronic components can be connected to Lego Mindstorms. Lego Mindstorms were sold as construction kits in a bundle including the brick, sensors, actuators and the software.

Other tangible programmable artifacts also use Logo derivatives, Fred Martin's Handy Crickets and the Cricket Logo language to program them, are one example. Cricket Logo's programming environment is text-based and quite close to the original Red Brick Logo. This design was used in other software to program smart physical objects as well. The Pic[13] based Handy Cricket system ships with electronic sensors and actuators (e. g. light sensors and motors) as well as with mechanical parts (i. e. gears). It is possible to use the Handy Cricket with standard electronic components, but a fixed resistor on the board design reduces the possibilities to connect sensors.

Design Example Logo: Programming the Physical and Visual Programming

Because we aim at bringing context into the programming environment we take a closer look at related programming environments for physical platforms in connection to constructionism.

Within the development of Logo languages the environment in which the programming took place, got more and more important. Logo itself is more an idea than a piece

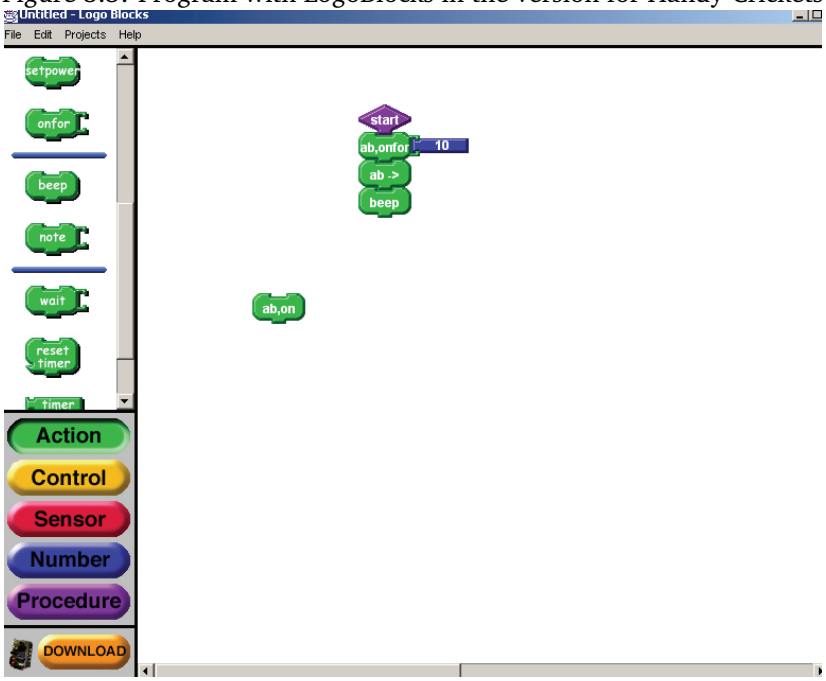
[12]The implications are associations with media images that can limit imagination. You can see that in our vignette. Amongst others, Reichel and Wiesner-Steiner discussed the topic in more detail (cf. [Reichel and Wiesner-Steiner, 2006]).

[13]http://www.microchip.com

of software and exists in different implementation often consisting only of a shell to enter Logo commands in (as for example in the well-known implementation UCB Logo[14]) and a space, where the output can be drawn. There are different implementations of Logo that differ in their additional features and *integrated development environments* (IDE). We already introduced the Red Brick Logo interface that is complex than most Logo environments. Most IDEs in the domain, e. g. for Cricket Logo or NQC[15] (a simple C-Syntax language for Lego Mindstorms), consist of a very basic Red Brick Logo-like interface with additional features e. g. syntax highlighting.

Begel [Begel, 1996] introduced a new way of programming the bricks using visual programming and the block metaphor in his Bachelor report in 1996. He wrote about the first version of LogoBlocks developed for the Programmable Brick. In LogoBlocks, one drags blocks labeled with a Logo command on a canvas to create a program. All blocks that are connected are interpreted as an interrelated program. The blocks are ordered in categories according to their functions, e. g. operators or sensors. In figure 3.3, you can see a program that switches on an output pin for ten milliseconds, inverses its direction and then beeps. Begel's work had a huge impact on development of applications in the area, being the model Lego used to create the Robotic Invention System. As main advantages of adding the visual element to the programming environment, Begel saw the lower entry barrier as well as the possibility to work with metaphors familiar to the user. In connection to the bricks the metaphor of assembling blocks (that look similar to Lego bricks) seems well chosen.

Figure 3.3: Program with LogoBlocks in the version for Handy Crickets



[14]http://www.cs.berkeley.edu/ bh/
[15]http://bricxcc.sourceforge.net/nqc/

When commercially available, Lego used the Robotic Invention System (RIS) Software in a bundle with Lego Mindstorms. In the first Lego Mindstorms' versions, they used Lego TS Logo. The RIS language is fully graphical and uses building blocks as a metaphor. It is inspired by and based on Begel's work. The metaphor sometimes breaks, e. g. in conditional statements, but in general blocks that are under each other are executed in this sequence. RIS is based on LogoBlocks, its concept of using building blocks as a metaphor is quite similar.

An important difference between LogoBlocks and RIS is the representation of the physical hardware within the visual elements. In LogoBlocks, the blocks were graphical elements with a text description for a command. The inputs and outputs of the physical hardware were also referred to by textual representation. In RIS on the other hand, most blocks refer to actions of special robot models. For example, the Acrobot is a robot children and young people can build with a manual and it has certain predefined methods, e. g. "dancing", that only work on an Acrobot that is assembled as the manual describes (we referred to problems that can arise through this design in our vignette). To program other robots or creations, RIS offers the "small blocks" that are similar to the blocks from Begel and represent single commands. Graphical dialogs that show iconic representations of Lego Mindstorms materials, e. g. sensors, guide the user through the process of configuring the blocks. Inputs and outputs are depicted with photographic representations of Lego Mindstorms' sensors and actuators in RIS.

The current version of the Lego Mindstorms construction kit is called *NXT*[16]. The software still uses the block metaphor, although it changed parts of the model and the references to pre-built robots are not in the focus right away, anymore.

In the development of Logo based languages, we can also find LogoWriter Robotics that is based on LogoWriter. LogoWriter Robotics is different in its basic languages and it is still in use—now it serves as underlying language for the commercial Microworlds[17]. Microworlds combine the Logo like programming language with a visual language and allow to embed simple communication between Microworlds, e. g. a little game, and the physical brick.

Design Example Logo: Programming and Complex Systems

Another extension of the Logo idea that we want to introduce only briefly focused on exploration of emergent phenomena. Resnick's Starlogo [Resnick, 1994], for example, aimed at allowing children and young people to learn about parallel processing and self-organizing structures. The users defined the behavior of the actors in a Logo language. Starlogo was the first attempt to use the Logo idea in the area of complex systems. Nowadays, the precursor Netlogo[18] is much more popular, newer versions of Starlogo are only available for Macintosh.

[16] http://mindstorms.lego.com/
[17] http://www.microworlds.com
[18] http://ccl.northwestern.edu/netlogo/

Design Example Logo: New Materials

The newer generation of educational physical hardware starts to experiment with different physical materials. Smart textiles are smart materials that is why smart materials are linked to our work.

We could see that from the programmable bricks which allow building many different applications, a focus on robotics emerged. Because robotics might not be interesting for all children and young people, some newer construction kits explore physical materials in different domains. We can find construction kits for e. g. smart jewelry [Dekoli and Mikhak, 2004] or smart textiles [Buechley et al., 2006] that aim at gaining girls as a target group.

There is also a huge progress in material development towards materials that are smart. We use the term *smart materials* in line with smart textiles (see the smart textiles section 5.2 for details) and the *ultra smart materials* or *third generation materials*. Smart materials therefore are materials that respond to their environment without being controlled by a micro-controller or electronic components. An example are *shape memory alloys* (again see section 5.2 for more details) that are used in constructionist learning environments (e. g. as muscle wire in robotics).

Eisenberg and his research group, called Craft Technology Group[19], play an important role in new materials and constructionism. The research group developed different educational applications of smart materials, e. g. the Programmable Hinge. In the Programmable Hinge, Eisenberg used shape memory alloys to display on film which parts of an electronic circuit were used at the very moment when a program was executed [Eisenberg, 2005].

Eisenberg's group and especially Buechley worked with smart textiles in constructionist education a lot. As the "materialists" of constructionist learning, they see a huge potential of smart textiles for education. Smart textiles are soft on the one hand, and a highly innovative medium on the other hand. That is a good opportunity to raise technology interest as discussed by Berglin [Berglin, 2005] and Eisenberg [Eisenberg et al., 2003], still there are only few applications in the educational domain. Eisenberg et al. [Eisenberg et al., 2006] argued that smart materials and invisible computing (and they regard smart textiles as a part of it) in the educational domain may be counterproductive, because the underlying principles of interaction are not transparent right away. That is why they should be used to make processes visible, to gain an educational value. The value of having a physical material in an educational context is according to Eisenberg et al. [Eisenberg et al., 2003] then:

> All sorts of physical artifacts might be seen as opportunities to convey subject matter content: a mobile might change its own configuration to illustrate ideas of balance or center of gravity; a fountain might display (or permit control of) complex phenomena in fluid flow; a terrarium might permit measurement of animal activity; a playground swing might be designed to convey ideas about the physics of resonance.

[19] http://l3d.cs.colorado.edu/ duck/Home.html

Programming in their theory has the role in

> Turning Ubiquitous Computing Artifacts into Means of Expression. [Eisenberg et al., 2006]

Through controlling the artifacts according to the learner's own ideas and interest they can be used as expressive tools. At the present date, we know about three approaches to use smart textiles for education and we would like to introduce them briefly.

A work by Berglin—the Spookies—used smart textile materials for computational toys that allow creative use and communication [Berglin, 2005]. The Spookies were figurative textile "dolls" that reacted to touch with changing colors or to input sound with output sound and that were able to communicate with other Spookies. Berglin's idea was that the soft and intuitive to use materials could help to provide an understanding and a tangible access to abstract ideas. This is the same concept that Eisenberg et al. (cf.[Eisenberg et al., 2006]) pointed out. Berglin's work used smart textiles, but not as a material children and young people use to design themselves.

In Eisenberg's research group, we find an application that implements the latter: Buechley's smart textiles (or *e-textiles*, to use Buechley's preferred term) construction kit [Buechley et al., 2007] that uses the textiles as construction material. Buechley et al. (cf. [Buechley et al., 2005], [Buechley, 2006], [Buechley et al., 2008]) developed an e-textile based construction kit which children and young people can use to build their own smart fashion clothes and accessories. On the technological side, Buechley combined traditional electronic materials with conductive fabric and the open source programming language and hardware Arduino. After developing mostly textile based PCBs (with an ironing-on technique which we will discuss in section 5.2), Buechley released the Arduino LilyPad [Buechley et al., 2008] as a commercially available standard PCB attachable to clothes through sewing through holes. The LilyPad uses a low-power consumption chip to match the requirements for wearables and smart textiles. Martin and Buechley also developed prototypes of a Handy Cricket with sewable holes—the Bling Cricket. Up to now there are no references or publications mentioning the prototypes, but some are in the possession of the DiMeB research group.

Buechley's approach seems to be especially appealing to girls, when the announcement for e-textile classes referred out the fashion application, instead of pointing out the technological aspects [Buechley, 2006].

Buechley [Buechley et al., 2007, 31] referred to the connection between smart textiles and everyday life in another direction than we argued, namely:

> A completed e-textile project is a permanent artifact that can be taken home and incorporated into a student's daily life.

We not only have a material that is familiar to the young designers, because of their practices around clothing, but children and young people can create an artifact and build new practices around it, because they are able to take it with them.

Another approach of using smart textiles in education, is from Lewis who runs a TV show called Switch[20] were smart fashion projects are explained with the objective to interest girls in technology by using fashion as a topic they find interesting.

Crafts and materials play a more important role in smart textiles in education, compared to educational robotics, although the focus of these pioneer projects that are using smart textiles for education is the idea to make technology appealing to girls using fashion as a medium. The computational and algorithmic side is seldom in focus in combination of smart textiles in education. Buechley's construction kit [Buechley, 2006] used plain Arduino as programming language. The Bling Cricket works with the regular Cricket Logo without changing the language according to the new medium the programs will run on. However, the researchers found the computational side important and argued for new or different programming tools especially when children and young people program new materials such as smart textiles.

> Even programming languages for children and young people might look somewhat different when those languages are tailored for ubiquitous computing artifacts. For example, one might create a "program browser" in which a student first selects a sample program based on the physical behavior that it produces; to pursue the example of wearables, our hypothetical student might look through a video library until he finds an example of a "light-up hat" that piques his imagination. [Eisenberg et al., 2006]

Design Example Logo: Tangible Programming

We mainly discussed Logo dialects where programming takes place on screen, as an action of an individual, and the program then runs on virtual or physical—maybe even smart—objects-to-think-with. The bricks were an example of this kind of digital artifact. We mention another approach to distinguish between programming tangible artifacts and *tangible programming*, because the terms are easily confused. In tangible programming, many implementations are based on Logo and the block metaphor, still it is different, because the tangible objects are used for programming. At least, when we use the term connected to McNerney and not connected to Elliott [Elliott, 2007] who used the term *tangible function programming* for programming with gestural interaction. McNerney's Tangible Programming Bricks represent Logo commands as well as variables and operators as physical bricks that can be plugged together [McNerney, 2000]. The programs created in that way will be attached to an artifact, e. g. a robot. The robot will then execute the program created with the physical blocks. As main arguments for tangible programming, one can see the advanced possibilities in cooperative programming, because a greater number of people can manipulate the physical blocks at the same time.

Conclusion Logo Family: Learning and Creating Meaning with Logo Programs

In the classical Logo and the floor turtle or turtle graphics, the drawing of the programs a child writes serve as the objects-to-think-with. These objects are external in the sense

[20] http://iheartswitch.com/

that other people can look at them and gain an understanding what they do and maybe of the creation process. The children and young people can think in drawings with the turtle ("identify with the turtle" [Papert, 1980, 56] as Papert called it) and in the concept of moving, as moving is a very familiar concept for these children and young people. It is easier to think in such concepts rather than in—abstract—mathematical constructs. These alternative concepts also make it easier for the learners to express themselves about abstract concepts.

> In Turtle work an almost inexhaustible source of "similar situations" is available because we draw on our own behavior, our own bodies. So, when in trouble, we can play Turtle. [Papert, 1980, 64]

Papert relied on familiarities in the concept of Logo. Children and young people know "moving", because they do it all the time in their real-world context.

The first physical output devices for Logo, for example the floor turtle, have the same functionality as the turtle graphics, only that they serve as an output device that is tangible and that acts in the physical world. The Programmable Bricks in their initial idea even go a step further. The activities Sargent et al. proposed, are much more connected to the children and young people's life-world. The brick does not act as an output device only—it can be used to create solutions for problems children and young people really have and care for. We see the flexible possibilities of bricks and their connection to the world as a great possibility.

> The Programmable Brick provides rich connections to the world. [Sargent et al., 1996, 162]

This connection makes it much easier for the learner to interpret individual meaning from interaction with the brick. The creations made with the bricks are also shared physical artifacts that act in the world. Other people can not only see a finished drawing as in the Logo case , but can interact with the artifact. That can ease creation of a shared meaning.

In our experiences with programmable bricks we can affirm these assumptions to some extent. We will discuss an example. One of our workshops (see section 6.1.1 for details on the general concept of our workshops) involved telling an adventure story with programmable bricks. The twelve children and young people that took part made up a story with quite a complex plot, because each child wanted her idea to be part of the common story and no ideas were left out. In the story a researcher tried to find Atlantis. On his way to the mysterious city, he had to solve several quests. There was door with a secret code he had to enter by pushing some buttons in special order. He had to dodge spiders moving from the ceiling and many more similar tasks.

Within the community who really worked on the project (the children and young people and their tutors that participated in the workshop) a shared meaning seemed to evolve. Each child knew about the part her object played within the story and through talking about the common story—creating and planning the plot—they learned about the others' projects as well.

However, the visitors of the final presentation did not grasp the story. They understood how they could interact with the objects , but missed the wider context—the story that glued the objects together. By looking at this example, we can see that the shared experience when creating the artifacts had an influence on the shared context. Sharing the creation process facilitates a common understanding compared to only interacting with the external artifact, so we should aim at a common creation in a constructionist learning environment.

3.8.4 Design Example: The Smalltalk Family

The Smalltalk family is closely related to constructionism. Pioneer Kay not only worked together with Papert (and still does in the *one laptop per child* (OLPC)[21] project), he also shared Papert's theoretical assumptions.

> With Dewey, Piaget and Paper we believe that children 'learn by doing'... Papert's kids need to do multiplication to make the size of their computer-drawn animation change. They have something to 'do' with it. [Kay, 1972, 5]

Kay and his colleagues combined the idea of a simple language with visual programming by adapting Smalltalk (which Kay developed as project leader), an object-oriented language, to create Squeak [Ingalls et al., 1997]. Squeak not only combines visual with text-based programming, it also deploys a paradigm for modeling GUIs called *Morphic* that is different from the standard Model-View-Control concept. Kay's approach was also inspired by Logo and the Logo ideas can be found in a lot of Squeak applications, for example in Etoys[22]. Etoys is a Squeak-based learning environment where programming is done with simply commands as you now it from the Logo family. Graphical objects can be expanded by attaching Squeak scripts to them. Through the combination with media objects, Squeak can be used to program object-oriented. Squeak was adapted to be used with Lego Mindstorms[23], but there is not a lot of development in the direction of physical platforms. The most popular example of a educational programming language written in Squeak, is the recent language Scratch that we will discuss in more detail later on.

Some concepts in the Smalltalk family focused even more on the interface aspect (in the sense of *graphical user interface*(GUI)) of programming and visual programming. DiSessa [DiSessa, 2000] argued towards creation of programs, but against writing source code and suggested his programming environment Boxer as a mixed educational language with a strong focus on visual programming as a very useful tool for this concept.

ToonTalk[24] by Kahn [Kahn, 1996] also emphasizes the visual direction. The programming environment was inspired by computer and video games; the user interacts in a fully virtual world with virtual characters. To write programs, the user manipulates a toolbox with some icons. For example, to delete items, one can use the vacuum cleaner.

[21] http://laptop.org/
[22] http://www.etoys.com/
[23] http://wiki.squeak.org/squeak/2412
[24] http://www.toontalk.com/

The most current version ToonTalk 3 includes online documentation. ToonTalk adds the idea of making the execution of a computer program visible.

Design Example Smalltalk: AgentSheets

We introduce AgentSheets as a precursor and also an inspiration for Scratch and as an example of a programming environment that focused on sharing and integrated sharing deeply into the environment. AgentSheets is a language and programming environment for children and young people (but also for adults) which allows the learner to program agents in what the creators of the language called *tactile programming*. The user creates objects and then specifies their behavior with a rule-based script language called Visual AgenTalk [Repenning et al., 1999]. Afterwards, she can run simulations based on the agents. We see that the very idea of AgentSheets is closely connected to StarLogo that we introduced in section 3.8.3. What inspired Scratch was another key though of the AgentSheets designers: the collaborative use. One of the main ideas behind AgentSheets was that

> programming environments that are easy to use and expressive must contain mechanisms that enhance the user's ability to comprehend programs and program fragments, to compose complex programs from simpler primitives, and most importantly, to easily share programs and program fragments within a community of users. [Repenning and Ambach, 1996, 102]

AgentSheets uses different "worlds" as metaphor, sharing took place in the *collaboration world* [Repenning and Ambach, 1996] in the early versions. In the current version of AgentSheets (2.5.1), this collaboration world is not present, but sharing functionality and connection to the Internet exists. Agents can use Google's image search for example. AgentSheets was ported to Lego Mindstorms [Gindling et al., 1995], but without transferring the idea of sharing through the Internet.

Design Example Smalltalk: Rich Media Creation with Scratch

Scratch is a novel programming language mainly meant for manipulation of media content implemented in Squeak. The main objective of the Scratch environment is to support disadvantaged young people and introduce them to *computer clubhouses*. We introduce the computer clubhouses here, because Scratch's development is closely connected to the approach and the software was developed with computer clubhouses as target audience in mind. Computer clubhouses is term coined by the MIT to describe open environments children and young people can join after school to work on computer-related projects (roughly comparable to a German Jugendfreizeitheim). They are supported by adults and also by their peers[25]. An important idea of computer clubhouses is that they are based in the local community and provide access to technology and knowledge that would otherwise not accessible for the young people. The computer clubhouses are especially situated in underprivileged areas.

[25] http://www.computerclubhouse.org/

The idea of Scratch is to take up social practices from the young people's life-world as an entry point to technology understanding and especially to programming. Here the entry point—the familiarity, as we called it before—, is the media content that plays a big role especially in the life of teenagers and creation and manipulation of media files is an activity young people are engaged with anyway (cf. [Maloney et al., 2004]). According to Maloney et al., there is a *Photoshop culture* among the young people: They creatively manipulate media with graphic manipulation programs, but they rather not program or create interactive media. Scratch enables young people to create animations based on their videos or games where a certain amount of programming is required.

Scratch also supports manipulations in the physical world by reading sensors and communicating with Lego Mindstorms. The main hardware platform is the Scratch sensor board[26]. The board has a fixed set of sensors (i. e. a light sensor, a sound sensor, a slider and a button) as well as some alligator clips that are connected to the board to measure plain resistance (and therefore allow to connect any custom-made sensors). There are some actuators on the board (light emitting diodes (*LEDs*)), but the board is made to work with sensors and translate the users' actions in the real-world into the software. One could also say, the board serves to control the programs one created with actions in the real-world. To transmit signals to the computer, the board is connected to the computer with a serial cable.

Different educational languages inspired the development of Scratch. We would like to point out the connection to block programming that Begel [Begel, 1996] used for LogoBlocks and Lego used for RIS. Scratch sticks to the block metaphor and labels the blocks with names of commands i. e. *move 10 steps*. Other inspirations for Scratch are—according to the authors—languages that allow sharing on the web as Maloney et. al. [Maloney et al., 2004] pointed out. As an example of such languages, the authors referred to AgentSheets.

Scratch is connected to a Internet-based platform called ScratchR [Monroy-Hernández, 2007] through different ways: a webserver and library server hosting a Scratch Object Library [Maloney et al., 2004]. The Scratch website or Internet platform is the place that allows sharing and publishing of the users' projects. The design is a weblog (blog) (based on the open source Movable Type Publishing Platform[27]), projects are ordered in chronological order and can be commented as well as rated and tagged by readers. It is also possible to share one's projects in thematic galleries. Because one can create Java applets with Scratch, the projects can be seen and tested or linked to directly in the web browser. The blog also offers extended social network features namely connections to other users by a "friendship" functionality.

The uploadload and download kind of sharing is integrated into the Scratch IDE. It is also possible to download and open other people's projects using the open menu of the IDE. In Version 1.0.1 it is not possible to search for other people's project. Scratch saves the children and young people's projects in binary format.

[26] http://scratch.mit.edu/scratch-board/
[27] http://www.movabletype.org/

The help system consists of HTML pages downloaded to the client computer and a forum and documents on the website. The authors argue from a constructionist background and support the argument that sharing is important in learning about programming. Scratch also has a connection to Maeda's idea of *creative coding* [Maeda, 1999]—using programming to create creative projects in media art—and therefore to programming languages like Arduino and Processing (see section 6.3.2 for details on the Processing family).

Scratch proposes sharability on different levels, from objects children and young people program, to the whole animation. The object-to-think-with can also be an object in the sense of *object oriented programming* (OOP), up to programs or projects.

Conclusion Smalltalk Family: Learning and Creating Meaning

The creators of Scratch aimed at teaching programming, according to Maloney et al. [Maloney et al., 2004]. They place Scratch within a context they consider interesting for young people namely the context of media arts [Peppler and Kafai, 2006]—inspired by the children and young people's interest for music videos. Programming therefore serves as a tool for self-expression. While media arts are the domain in which the young people create applications, Scratch also places the language in a real-world community setting: the computer clubhouses.

Scratch allows creating objects that are external, sharable and visible. The artifacts children and young people create are also "cool" what is important in the young people's world. These objects-to-think-with are the animations or games created with Scratch, but can also be objects on a lower level: a media object, e.g. a song, or a programming object, e.g. a new class.

When going back to the different extensions of constructionism, Scratch can serve as an example of distributed constructionism, but is also inspired by the idea of constructionist cultures, because the Scratch creators try to place Scratch in the real-world community environment of computer clubhouses.

We can say that Scratch takes up a domain interesting and familiar to its target group and supports sharing, not only as a supportive context, but also to make the objects-to-think-with widely visible and accessible.

3.8.5 Design Example: MUDs and MOOs

MUDs (multi-user dungeons) are text-based role-playing adventure games played on the Internet. A *MOO* (MUD object oriented), as shortly mentioned before, is a virtual space where the users create their stories, characters and the environment they play in, by using a simple programming language. MOOs are based on MUDs. The main difference is that MUDs are object-based, respectively object-oriented.

In constructionist learning designs, MOOs were taken up as idea to build (virtual) communities that perform construction activities together. A special purpose MOO, made to support learning for children and young people, was realized in MOOSE Crossing (cf. [Bruckman et al., 2001], [Bruckman, 1997]). Children and young people were enabled

to create characters online in a text-based language; the characters could interact with each other in the MOOSE Crossing environment. To go to a place, join another character and describe one's character, one could for example type in:

Listing 3.3: Basic interaction with MOOSE Crossing

```
1 tel ``MZH Lab''
2 say ``Hi, I am Milena''
3 join ``Eva''
4 describe me as ``tired-looking and grumpy''
```

The idea is to have a very simple text based language (clearly inspired by Logo) to construct environment as well as adventure.

Another implementation of MOOs in the constructionist context is the MediaMOO (cf. [Bruckman and Resnick, 1995]). It was created for students and professionals to build and support distributed professional communities.

MOOSE Crossing and MediaMOO were the first constructionist applications that were completely web based and virtual. However, especially in the MOOSE Crossing scenario, Bruckman argued for embedding the virtual environment into a real-world community [Bruckman, 1997].

The MOOSE Crossing environment is a constructionist learning environment, because the players actively create the whole adventure themselves and learn basic programming as well as to express themselves in written language. The environment in MOOSE Crossing is community-based, because the virtual space is shared with all players and one player communicates with another. All players that are taking part in the adventure, also create the story together.

In MediaMOO, the whole idea is creating and forming a community. The members create the whole environment in which they meet and communicate themselves.

Bruckman emphasized on the community aspect of MOOs. People use the virtual environments for fun and sociability. The space the virtual community creates is therefore called *third place* by Bruckman, a notion we also find in other community based constructionist communities as well. The term third place is coined by Oldenburg [Oldenburg, 1989] who referred to neutral places for informal social interaction with it. According to Oldenburg, examples of third places are coffee shops or bars. Bruckman saw MediaMOO as a third place, because people go there for conversation [Bruckman and Resnick, 1995]. The MOOs are, again, an example of distributed constructionism as well as for constructionist communities.

Conclusion: Learning and Meaning Making in MUDs and MOOs

So what are the objects and why are they personally meaningful? Especially in MOOs everything is an object. A character, a room, objects in the room, the other players... Users can create their own rooms and their own objects of choice. Through this they relate to what is important to them and therefore each object is personally meaningful.

Bruckman studied the MediaMOO (as well as MOOSE Crossing) in more detail and found out that people mainly modeled their real-world environment as a space in the

virtual community. For example, users did build the MIT with its various labs, e. g. the E and L Garden, or other places the users worked in, for example the Apple R and D Lab (cf. [Bruckman and Resnick, 1995]). These models are very close to the places as they are in reality, describing the space with spatial metaphors.

In the MediaMOO project [Bruckman and Resnick, 1995], Bruckman and Resnick described their experiences in more detail and reported about a user who was confused and lost in a MIT building, until he met a girl from the company, he worked at. She was showing him the building of the company they both worked at, and he felt at home at once, recognizing the companies' building in the textual description. The example served to show how powerful text as an expression can be in Bruckman and Resnick's essay. We read it as an example of meaning creation in a MUD. The user cannot make sense out of an unfamiliar building. Then he meets someone with shared experiences and they understand each other, because of this shared context. The environment modeled according to the real-world building the user is familiar with, works well for him. In this respect, the MediaMOO is closely connected to the real-world, because it is modeled very close to physical reality (e. g. using a spatial metaphor and modeling buildings as they are) and therefore draws on the shared experiences in the real-world. A MUD like MediaMOO is very different from typical MUDs that are adventure games taking place in fantasy worlds[28].

MOOSE Crossing is more similar to the adventure game MUDs. MOOSE Crossing is organized in different *rooms* that provide "Contexts for social interaction" [Bruckman, 1997, 23]. These rooms are often fantasy places (Bruckman referred to pools as rooms, for example Jack's Swimming pool). The rooms are clearly inspired by the real-world of the children and young people, but add fantasy objects and possibilities, e. g. swimming with dolphins or talking underwater.

In MOOSE Crossing support and introduction of the system for *newbies*—new and still inexperienced users—is usually done by the members themselves. If the users are lucky, they meet someone in the environment they share interests with. Others are less lucky and do not meet other people online and get confused and frustrated with the system (see for example [Bruckman, 1997, 119]).

Research on MOOSE crossing was conducted in after school programs or at the research lab. A real-world interaction supports the building of virtual communities according to Bruckman [Bruckman, 1997].

3.9 Design Concepts: Design Process

We mention design methods very briefly at this place, to point out some experiences researchers made in connection to constructionist learning and programming environments, and discuss Druin and her colleague's more advanced methods on the design process with children and young people in section 5.3. Participatory design methods or designing and testing with children and young people play an important role in the de-

[28] MUDs are extensively studied by Sherry Turkle, see for example *Life on Screen* [Turkle, 1997].

velopment of constructionist learning environments. Resnick and Silverman, who summarized their experiences on designing construction kits for children and young people, concluded that the designer should:

> Give people what they want—not what they ask for.[Resnick and Silverman, 2005, 117]

At first, the statement seems to be a provocation, but the authors clarified that they found it rather useful to observe users interacting with prototypes than to directly ask them what and how to design. In their experience, especially young children often came up with ideas difficult to realize when asked directly:

> Elementary-school students recommended that we design the Bricks so that they could fly. [Resnick and Silverman, 2005, 120]

Although we find different approaches of methods to structure the design process, the constructionist learning environments are all designed in very close contact to the target audience—the children and young people.

3.10 Design: Summary

In section 3.8, we looked at design examples of programming environments for children and young people. By slightly adapting Guzdial's taxonomy, we came up with three main streams of constructionist learning and programming environments: We looked in more detail at

- the Logo family
- the Smalltalk family
- MUDs respectively MOOs.

The three families are constructionist learning environments, because they were developed with constructionism in mind, enable children and young people to create programs themselves and were developed by a group of people that know and worked with Papert. For each of the three main directions, we discussed one environment in more detail: Programmable Bricks, Scratch and MOOSE Crossing and MediaMOO. We gave a very brief overview how the directions developed to show what ideas the designers had in mind when designing the environments.

When we relate these examples to constructionist theory and its extensions, all examples show how a personally meaningful object is created and what the child or young person constructs. Scratch serves especially well as an example of distributed constructionism while MOOSE Crossing and MediaMOO implement a constructionist culture.

In all examples, constructionist learning is supported, because the children actively build and design an artifact—a Logo program, a media animation or an adventure or room. For the examples, we always asked how the process of creation of meaning was considered by the designers. The question on creation of meaning addresses the second

assumption constructionism is based on—the object children and young people work on should be personally meaningful. The Programmable Bricks enable the children and young people to create projects that have an affect in their real physical world, which is why children and young people can easily create a project meaningful to them. For the design of Scratch, the designers took a look into the practices of their future users and identified a domain the young users find interesting: media arts and music videos. By taking up the real-world practices, the designers allow the users to create something meaningful. MOOSE Crossing and MediaMOO are very flexible; the users create the environment themselves. That is why the users can take up something they care for and make it their personally meaningful object-to-think-with.

When we look at the constructionist learning environments with our contextual perspective in mind, we can see the following: On the one hand—we saw this perspective in the constructionist theory—the learning environments try to place or situate the technology into a (social) context. On the other hand, we could see that the designers tried to take up and draw on the user's context: They take up meaningful practices, and use metaphors the user is familiar with. We can learn for our concept that both directions are necessary: bring new technologies into a social context and take up social practices to design new technologies.

In the next chapter, we aim at bringing together the design implications of the contextual perspective with the theory and design implications of constructionist learning environments.

Chapter 4

Towards a Concept: Bringing Together Context and Constructionism

4.1 Context and Constructionism

In the first chapter, we discussed the concept of context relating ourselves to approaches based on phenomenology. In the second chapter, we took a closer look at constructionism and constructionist learning environments—implementations inspired by the theoretical background.

Now, we would like to point out similarities of both theories. In phenomenology (see section 2.2.2 for details), especially when we relate to Husserl, we have seen an emphasis on the individual who creates meaning through her interaction with the world. This whole world the individual acts in, is her context that influences the way she interacts with digital artifacts and this context is unstable, as it emerges in a situation. Individuals create meaning in an ad hoc situation. Creating shared meaning takes place through communication in Husserl's approach. For Schütz, the individual's meaning creation is not purely individual, he assumed we share basic assumption on how the world works among people. The level of shared understanding depends on shared experiences. We will rely on Schütz' understanding of intersubjectivity in this thesis.

In constructionism (and already in Piaget's constructivism—see chapter 3 for details), we find a similar basic assumption: The individual—in this case our learner—creates a concept of the world by interacting with it and in it. This concept can change, for example, during a construction activity, when it does not work anymore. Papert added the assumption that creating a personally meaningful external object, is a very good way to challenge the learner's concept. These two points: An emphasis on the externality and an emphasis on the personally meaningfulness of objects are the main ideas behind Papert's constructionism. According to Papert, during the design and learning process

the designer—who is the learner at the same time—will take up meanings and bring them into a new context, resulting in a new object.

The creation of shared meaning or creating a common context is less discussed in constructionism compared to the individual's learning process. We find different extensions to constructionism partly inspired by Vygotsky that suggest sharing of created objects as well as building communities around construction activities as ideas to bring a social component into Papert's constructionism. Even though these new extensions do not explicitly use the notion of context, we can use our contextual perspective to summarize these trends in constructionism under a common umbrella. What the distributed constructionism as well as the constructionist culture and social constructionism perspective have in common, is that they rely on shared experiences and on creating a shared province of meaning, to say it in Schütz' words. These shared experiences and through this a common context can result from interacting with the others' objects as well as constructing together in a shared real or virtual space. The social communities around construction activities situate the learning process into a real-world situation and a real-world community. The meaning making process is shared and situated in a (social) context.

We can see that the context notion inspired by phenomenology fits well together with constructionist learning theory; the theories share many basic assumptions by considering the individual in her context as most important. For a concept of constructionism in context, we can summarize:

1. *The individual's context* depends on how she interprets an action situated in the current situation depending on her experiences and motivations, but also on an ad hoc constellation of human and non-human actors. The object-to-think-with is personally meaningful, because it has a meaning for the individual in her context. The learning process takes place through designing and making the object.

2. *A shared context* always exists on the low level of basic assumptions on how the world works (and in the case of children and young people also depends on their age). Shared worlds or shared provinces of meaning that lead to a shared context can arise from living the same experiences. These experiences can be, for example, sharing of objects, or performing construction activities together. Constructionism emphasizes another advantage of creating together, namely that shared construction activities also provide help and that sociability is simply motivating and more fun for the young learners.

In chapter 3 on constructionism, we ended with pointing out how the chosen examples of learning environments implement a constructionist approach. We mentioned design decisions and how the environments are designed so that the users understand them, briefly. In the following sections, we aim at combining the design suggestions, such as Dourish gave with his embodied interaction framework, with the design decisions resulting in the constructionist learning environments which we introduced in the last chapter. We will examine how the design suggestions can influence constructionist learning environments or how we can interpret them under such a perspective.

4.2 Designing Contextual Constructionist Learning Environments

We discussed familiarities as well as affordances, appropriations and flexibility as rather general concepts to help the designer to create digital artifacts that can be used in and adapted to different contexts in section 2.3.1. We also discussed methods of social computing to change the design process of digital artifacts and to involve the user in her environment from an early stage on. When looking at the examples of constructionist learning environments, we saw that these general concepts in interaction design also played an important role in the design of constructionist learning and especially programming environments. The designers of the programming environments often did not discuss these general concepts as such. Therefore, we will show how the concepts are implicitly used in the examples of the environments which we discussed in the last chapter.

4.2.1 Familiarities

In the first step, we look at the concept of familiarities and point out how the designers use familiarities in their design. In constructionist learning environments, we saw examples how successful environments were designed and also how the designs developed over time. When looking at these environments, we can see that familiarity is a concept that often appears when design decisions are discussed though it is not necessarily referred to as familiarity. Starting with Papert's turtle graphics, the turtle moved on the screen, because Papert wanted children and young people to understand the turtle's action by drawing to their familiar experiences with moving in the world. The Programmable Bricks go even further as they are aimed at really being a part of the children and young people's life-world. By not only referring to familiarities, but integrating the bricks into the physical world, they can be connected to real life easily. Here, the constructionist learning environments rely on very basic familiarities, movement is a metaphor that is probably understandable for all children and young people.

The first evidence of drawing on familiarities in the design of programming languages we find in Papert's Logo, and again, when looking at Begel's block metaphor in which he tried to take up something familiar. In tangible programming, we can see similar considerations: Programming moves into the physical domain, because people know how to act there. The Tangible Programming Bricks from McNerney [McNerney, 2000] even relied on the same metaphor as Begel [Begel, 1996] did—assembling building blocks.

Scratch uses familiarities differently. The designers draw on activities—consuming and creating media in form of videos and animation. These are activities of a certain group—young people. Scratch's designers then used this interesting application domain to bring in the learning of programming through the back door. For their concept on how to include sharing of objects, they did the same and relied on young people's experiences with web-based *social networking sites (SNS)* (cf. [Monroy-Hernández and Resnick, 2008]).

In the MUDs and MOOs example, the language and architecture allows building virtually everything. By using spatial metaphors, for example buildings and navigating through different *rooms*, it still draws on shared experiences and familiarities of its users. In the examples Bruckman and Resnick [Bruckman and Resnick, 1996, 207-221] studied in their research, we can see that people build familiar things within the MOOs, e. g. the buildings in which they work in real life, and then understand these creations depending on what they know (where they work or where they have been before) from physical reality.

4.2.2 Flexibility and Appropriations

We used the term flexibility to refer to digital artifacts that are able to adapt to different contexts and allow multiple interpretations. In a constructionist scenario, we can differentiate between flexibility in the usage of the construction kits or tools—the applications or objects children and young people can make with the tools—, and the flexibility in the design of the tools themselves (for example, does a child need the full range of a general-purpose programming language to program a brick?).

Referring to the design of tools (in particular to the design of menu panels in Scratch), Resnick and Silverman concluded in their essay on their experiences with construction kits that:

> Often, designs with well-chosen parameters are more successful than designs with fully-adjustable parameters. [Resnick and Silverman, 2005]

On the other hand, an important design principle in constructionist learning environments from the Logo language is to design for a "low floor and high ceiling" [Resnick and Silverman, 2005, 117]. The principle means to allow an easy entry for a novice, but a great range of possibilities for an expert user. Resnick and Silverman adapted the principle and added to design for "wide walls" to their recommendations on the design of construction kits. Design for wide walls referred to a wide range of applications that can be built with the construction kit similar to the initial design of the Programmable Bricks. The Programmable Bricks were designed for a wide and flexible use as the essay *Twenty Things to Do with a Programmable Brick* [Resnick, 1993] implied. Here, flexibility played an important role.
When designing for such a flexible use, one automatically designs for appropriations. For example, finding new and unintended ways to use a Programmable Brick is always an appropriation.

4.3 Principles

Now, we have shown that there was always a relation between the general design concepts discussed in interaction design and embodied interaction at the moment and the design of constructionist learning environments. By discussing the relation in detail, we made it explicit. Now, we can derive some design "principles" that help designers of

constructionist learning environments to start the design based on the general concepts. Through this, we believe context can be brought into constructionist learning environments from the underlying architecture on. As example how such an architecture can look we designed, implemented and studied the EduWear construction kit (see chapter 6 for details).

To summarize our findings, we conclude that to design a constructionist learning application that implements *constructionism in context* following principles should be kept in mind:

1. *Start designs from familiarities:* We saw that, for example in Logo, the design started from a familiarity—moving. This is a familiarity for all people. In Scratch, the designers took up the Photoshop culture, the practice of children and young people in computer clubhouses to manipulate digital media content. The practice is familiar to a certain group of people only. That is why the first step of designing with familiarities is to identify them. The central questions are: Who are we designing for and what are these users interested in? How do they use the things they are familiar with? Observations and analysis of the social practices children and young people developed around technologies, but also other things are helpful. Working with *participatory design* methods can be a way to identify these interests as well as the real social practices that evolved over time. That is why we also suggest to:

2. *Develop in a participatory design process:* In a participatory design process, we work directly with the future users of the technologies we develop. We can study practices with existing technologies as well as with prototypes of new technologies what is essential to identify familiarities and see how users interact with the digital artifacts in context. Such a participatory design process can have different faces. One can use special participatory design methods adapted for using them with children and young people e. g. *cooperative inquiry* (see section 5.3 for more details), or—as Resnick and Silverman suggested—design iteratively with prototypes in many iterations and observe the users interacting with the prototypes. The important thing is to gain information about the real use of the artifacts in the users' contexts.

3. *Create a shared context and situate the learning process in there:* We have seen that the creation of shared meaning depends on communication and on shared experiences. Social learning and sociability are important for plain fun as well as to provide a supportive context and to connect new knowledge to a wider context (for example, in the way Papert described with the samba schools). Therefore, we should design for shared experiences and situate the construction activity into a community or learning scenario. To use familiarities in design is a way to achieve that. Another way can be creating real and virtual constructionist communities and share objects-to-think-with. Regarding the question on how to design for communities, we find helpful suggestions in the constructionist learning environments we looked at in detail.

4. *Create (third) places:* We have seen that a virtual community is more likely to be successful, when they have their own place—a third place—where they go for fun and sociability. From a constructionist approach, they should even design their own place so that the place can serve as object-to-think-with. When the users create the place themselves, they can most likely draw on familiarities themselves. That is why we would like to add the next principle.

5. *Allow the communities to create and design their own place:* The MUDs and MOOs are only one way to allow users designing the environment instead of designing "only" the object-to-think with. We will explore other possibilities in the next chapter.

6. *Consider different aspects of the shared context:* In the principles three, we suggested to create a shared context for example a community and in principle four that this space should be a place. Community building alone will not necessarily result in a supportive context as well as technology designed for flexible use might not come to a full potential in a school lesson, where children and young people use it to solve a given exercise. We used a wide notion of context to include a range of actors. The setting the construction activity takes place in has a huge impact on the learning process. When aiming at creating such a place, the setting and its different influences have to be considered.

7. *Bring contextual information in the environments:* As the meaning making process depends on the shared experiences, it seems important to bring in as much information of the context the objects were created in as possible. This information is important, for example, when sharing them on the Internet. Of course—as we pointed out before—meaning creation depends on the individual's interpretation to a great deal, but information can help as it can draw on shared experiences. By describing a robot and adding at contextual information that it serves as an actor in a role play, others might connect to their experiences of role-playing and might understand why the robot is dressed in a funny costume.

8. *Make the programming environment a place and draw on familiarities:* In principle three to seven, we pointed out the importance of creating places and what to consider when designing them. We showed how communities use third places to socialize and exchange experiences as well as creating their own places. Now, we enlarge the argument and include the programming environment as a part of the place. In the MOO example, the environment and the object-to-think-with are the same thing. We argue that this should be similar when designing programming environments: They should draw on familiarities and be a place to socialize as well. The visual programming languages made a step in this direction by using metaphors the users are familiar with. For example, Scratch added features from social web applications to support sharing and sociability.

9. *Design for wide walls and for flexibility:* We borrow Resnick and Silverman's notion of wide walls here that describes flexible designs for different application domains.

It seems that even when we draw on familiarities, we should aim at giving children and young people the space to be flexible and to create very different applications. That is only one part of flexible designs, we suggest. To make the digital artifacts adaptable to different contexts, we go a step further and aim at making the programming environment more flexible as well.

10. *Make something happen in the physical world:* Not all applications that consider context and not all constructionist learning environments have a direct impact on the physical world. Still, for example, tangible artifacts as the bricks demonstrate that through placing computing into the physical domain, it is very easy to draw on familiar practices. Therefore, we consider a strong connection to the physical world when creating constructionist learning environments as important.

11. *Carefully adjust between flexibility and familiarity:* We placed emphasis on the importance of both flexibility and familiarity. However, relying too much on familiarities might confine flexibility, as we have seen in our vignette. The design of the software relied on familiarities—activities like dancing or dodging as blocks. The children and young people tried to use the blocks in a different context—another hardware model—and failed. On the other hand, a very flexible design can be very hard to use when the users have no experiences with other things they can draw on.

4.4 Relation to Our Vignette

When we come back to our vignette, our scenario from the beginning, we should ask now how our considerations and principles address the problems introduced in the story. The first problem we saw in the scenario, was Lena's boredom with the topic of robotics. By allowing her to create a project based on her ideas, it could become more meaningful to her, and maybe taking up a domain familiar to her—instead of robotics—might catch her interest. Even a flexible use of the brick—by building their own model—would have been a better solution. We see that technology design based on our principles could have helped, but the setting, the context of creation, also plays an important role. Here, the context includes the pedagogical concept, for example the time pressure that forbid to make a personally meaningful object-to-think-with.

The next problem arose when the children and young people started programming. The software used familiar metaphors—activities like dodging—, but it was not intended for the flexible use the children and young people tried. The software relied on certain programs for certain hardware, but the children and young people wanted to use all programs on their hardware model. By bringing in their context—in the sense of the hardware model or their idea—they could have been helped. In the programming situation, we also saw an example of an ad hoc situated meaning creation. The children built their model and imagined an idea what is was supposed to do. They found a block that seemed to fit reasonably well. It did not, what was not understandable for the children and young people until the tutor explained it.

We also saw that examples or support were not available when needed. Support from a tutor came too late, the children and young people where already frustrated. An offline or online community could have helped to address this problem.

The last problem our three children encountered, was that even during his short absence from the others' programming Tobi lost touch with the experiences the other children made. Information on why Lena and Marc changed their concept would have been a help for Tobi.

Chapter 5

Suitable Application Domains: Tagging and Smart Textiles

In the previous chapters, we discussed theoretical foundations and introduced our constructionism in context concept. We derived this concept from studying the theory behind the notion of context and constructionism as well as from discussing learning environments that are based on constructionism. Now, we will focus on the construction part of the thesis.

We concluded in our concept, amongst other points, to draw on familiarities, allow the users to design environments that are open and flexible themselves, bring in their own context and create places to socialize. We saw that it is possible to take up application domains interesting and familiar to the target audience of the learning environment. Scratch and its approach to use music videos as the topic of the learning environment was a great example of how to do it. The application domains and technologies should not only be familiar—they also have to be suitable to adapt to different appropriations. In our concept, we addressed the issue with recommending to "carefully adjust between familiarity and flexibility".

The aim of this chapter is to identify suitable application fields that can be the base for an environment that implements constructionism in context. We start by discussing two application domains in more detail: *social software*—and especially tagging—and smart textiles. We chose both fields, because they are familiar to our target audience of children and young people. Furthermore, both fields have a deep connection to sociability and flexibility and appropriations can be found in the domains. By discussing them further and showing how end users as well as designers work with them, we attempt at showing how the design principles relate to both fields.

To include the user in the design process from an early stage on, we also introduce research in participatory design for children and young people, focusing on methods Druin and her colleagues studied.

5.1 Social Software

We wrote in chapter 1—our introduction—that we consider social software as a possibility to ease the creation of a flexible supportive community. Such a community could address the problem of frustration the children and young people experienced in our vignette. The frustration was partly caused by the lack of a tutor, or an example, to support them with their work. Children and young people use the Internet for social interaction and communication anyways (we will show that in section 5.1.1), so we assume that the technology is familiar.

The questions we ask for social software are: How do especially children and young people use social software? How does social software allow them to relate to their context and how can they be part of designing the software?

Sociability is considered important in learning as we pointed out in section 3.2. But learning is not the only field where social interaction and communication play an important role. Research on *Computer supported collaborative work* (CSCW) environments emphasizes on the social. Also, many private users of the Internet use it for social communication. The Internet and its changes towards a mass medium for communication is often referred to as the *social web* [Wilker, 2007]. The term is used to describe different things—in our understanding an Internet that is fostering social functionality. From systems that support communication and collaboration, further development took place. Many current systems are open environments shaped by users through collaboration. Users create and shape their environments themselves as we saw it in MOOSE Crossing in section 3.8.5, instead of using the medium passively. This approach goes even further than *user-generated content*. User-generated content refers to content end-users added to a web-based system. The structure and navigation of that system can be fixed and made by someone else, though. One example of social software is the Internet or social web and its current development to user-driven environments in which users not only share content and create content and shape environments, but also connect socially. We would like to start with a wide definition of the term social software by Tepper [Tepper, 2003, 19]:

> Social software refers to various, loosely connected types of applications that allow individuals to communicate with one another, and to track discussions across the Web as they happen.

This definition includes many applications. Simple emails between individuals, for example. Coates gave more current examples in his definition, which is narrower, and referred to new Internet technologies only:

> Social Software can be loosely defined as software which supports, extends, or derives added value from human social behavior—message-boards, musical taste-sharing, photo-sharing, instant messaging, mailing lists, social networking. [Coates, 2005]

In his definition, the focus is on groups of people that communicate. Not the social interaction between two individuals is in the foreground, but the interaction of many to

many people as one can find it for example in a social networking site. We regard the SchülerVZ[1], popular amongst German youngsters as an example of a social networking site (SNS). When a user adds her current mood to her profile, e. g. "Just came back from school, feeling tired", all of her friends can track that information, not one specific friend only. Social applications focus on human-to-human interaction—the examples Coates referred to do so as well. The goal of these systems Coates referred to, might be a different one. Not an improved interaction is the main goal, but, for example, selling a product to the customer. Children and young people use social software seldom for e-commerce. In the next section, we discuss how they use social software in more detail.

5.1.1 Children's Use of the Internet, Tagging and Mobile Devices

Social software is very important for children and young people. When they use the Internet, amongst other current technologies, e. g. mobile communication devices, it is mainly used as a social medium and for the fun of sociability.

Large studies exist on how children and young people use the Internet (e. g. the JIM study [Medienpädagogischer Forschungsverbund Südwest, 2006], [Medienpädagogischer Forschungsverbund Südwest, 2007]). The JIM study is a study in Germany in which more than a thousand of young people aged between twelve and nineteen are interviewed concerning their media usage, annually. Comparable studies exist, e. g. for the North-American region. Internet is only one medium the children and young people are interviewed about; the study also considers other digital media as well as traditional media. JIM considered, for example, blogs as a part of a Web 2.0 section since 2006 (they were rarely used by young people in Germany in 2006). The main function of the Internet for the young people is communication; in 2006 chat and email were the most common technologies they used. In 2007, the behavior changed, the usage of Web 2.0 technologies increased, especially the use of SNS, e. g. SchülerVZ or the international SNS MySpace[2] and content creation and sharing sites like YouTube[3]. 9 % of the interviewed people used YouTube actively (produced their own content), 60 % used it passively, 17 % only heard of the name and 14 % never heard about the service before the interview. The usage pattern is very system specific. While MySpace has similar numbers as YouTube, 93 % of the interviewees did not hear about the photo-sharing portal Flickr[4]—one of the bigger Web 2.0. services owned by Yahoo—before the interview. Although a number of children and young people use what is called Web 2.0 according to the JIM study actively, the passive use (for example watching videos at YouTube instead of making respectively uploading them) is more common.

The JIM study does not refer to special technologies like tagging or using search engines in detail. When children and young people use systems like YouTube, we assume they are familiar with tagging, because the systems force the user to tag when uploading

[1] http://www.schuelervz.de
[2] http://www.myspace.com
[3] http://www.youtube.com
[4] http://www.flickr.com

content. Studies from other sources focus on how children and young people really use technology, for example search engines (see [Bilal, 2000] for more details).

The results of Bilal's studies were the following: Children and young people tend to browse rather than to use keyword search and succeeded in the given tasks more often when using an interface with links. These results suggest that children and young people might be able to use tags as tools for navigation, but may have problems (or little motivation) in tagging themselves, because forcing their ideas and concepts into keywords is not familiar. Special search engines for children and young people, e. g. Blinde Kuh[5] or Yahoo Kids[6], rely on categories, though Yahoo Kids uses a *tagcloud* as keyword visualization. However, the tags in Yahoo Kids are not user-generated.

Another interesting part of the JIM study about how young people use digital media—we refer to this part of the study, because it is connected to smart textiles—considered mobile devices. In section 5.2, we will see that mobile communication is one field smart textiles are used for. The Jim study 2007 found out that 94 % of the interviewees that are between twelve and nineteen years old, have a mobile phone and more than 80 % of their phones have a camera. The camera indicates that the children and young people own quite recent and up-to-date mobile phones. Text-messaging (SMS) is the most commonly used way of communication with the phones.

84 % of the young people also have and use MP3 players, what makes the MP3 player the second common device young people own. After mobile phone and MP3 player, we find CD-player, radio and TV before the computer (or notebook). The ownership of mobile devices differs strongly, for example, depending on gender and level of education, but mobile phone and MP3 player are devices the majority of young people own, even amongst the different groups.

The studies give a good insight into what kind of technologies children and young people use. Some information on *how* they use these technologies, in the sense of what functions or features they use (e. g. text-messaging is more popular than using the phone to make voice calls), is also given. The JIM study provided information on the topics young people associate digital media with. By far most important is "love and friendship" before "music".

However, from the studies we cannot get a deeper impression on the practices the young people build around these activities. What kind of messages do they send? Qualitative studies, for example by Boyd on SNS (cf. [Boyd, 2007]) or by Rheingold [Rheingold, 2003] on mobile phones, give much more details on the social practices around the technology. For example, according to Rheingold a lot of young people use text messaging for small notices to show their friend they are thinking about them, e. g. "thinking of you". Even sending a blank text message is a common practice.

> A young man sends the object of his attention a blank message or a bland one such as "that was a nice party." The recipient can choose to ignore the initiation or to respond and thereby signal interest. [Rheingold, 2003, 26]

[5] http://www.blinde-kuh.de/
[6] http://kids.yahoo.com/

Rheingold also pointed out that what we introduced with concept of appropriations plays an important role in how children and young people use mobile devices. Sending a blank message for dating purposes is definitely an example of such an appropriation.

We see that the Internet, even the Web 2.0, as well as mobile devices, are familiar technologies for children and young people. They use them as social technologies flexibly by creating their own appropriations.

To find out whether tagging is used amongst children and young people, we have to conduct more research. The JIM study implicated that some systems that apply tagging are widely used, while other studies showed that keyword search is not very popular for children and young people.

5.1.2 Social Navigation

In the domain of social software new social practices and technologies have been established. The boundaries between author and user blur and navigational structures, e. g. in the Internet, change. Users no longer only navigate on a path through a system that the designer completely designed and foresaw. Users browse the Internet in different ways and with different motivations. An example of how people navigate is social navigation, defined as "navigation towards a cluster of people or navigation because other people have looked at something" by Dourish and Chalmers [Dourish and Chalmers, 1994, 18]. In social navigation, people navigate through the Internet similarly to the way they navigate in the real-world. An example of real-world's social navigation is when one is searching for a conference room. One might see a lot of well-dressed people gathering in front of a door and assume this is where the conference must be.

An Internet-based example of social navigation is a *recommender system*. These systems analyze the user's behavior and recommend datasets, because of the current users and previous users' behavior [Konstan and Riedl, 2003]. The original definition by the creator of the term recommender system is the following:

> In a typical recommender system people provide recommendations as inputs which the system then aggregates and directs to appropriate recipients. [Resnick and Varian, 1997, 56].

A well-known example of a recommender system is the bookstore Amazon[7] that calculates recommendations for products. The output is a recommendation which Amazon announces in the form of "other people who bought the book you are looking at at the very moment also bought the following products...". People then will probably navigate to the recommended product. The recommender is an example of social navigation, because people navigate on traces of other people that used the system before them. Especially for commercial systems *social navigation* (see section 5.1.2) is a popular topic and we argue that social navigation could also be interesting for learning environments.

We introduce some basic concepts of recommender systems, because we will go back to them in our implementation details.

[7] http://www.amazon.com

A main difference in recommender systems is between *content-based* and *collaborative* or *social filtering* (cf. [Goldberg et al., 1992]). In content-based systems recommendations are calculated, because of the similarity of elements in a data set. The similarity can either be calculated by automated analysis of the content or by using attributes of the items entered manually, e. g. as meta-data. For example, in an online shop new items are placed manually in different categories, e. g. "Ladies" and "Shoes". These categories are then used to calculate similarities.

Collaborative or social filtering is more similar to the concept of social navigation. The user's behavior and how she acts on an item is analyzed to calculate recommendations. We already brought the example of Amazon's recommender that makes use of collaborative filtering to calculate recommendations. Both methods can be combined in *hybrid systems* (cf. [Burke, 2002]).

Other forms of social navigation may include systems that display their current users and what these users are currently looking at. For example, in some e-learning systems (cf. the open source learning system Moodle[8]) we can find a "users currently online" display.

Users are part of designing and creating the virtual place or space through their very actions. That is how the technology relates to our design principles: users create and design their own space. When someone buys a book at Amazon, she also influences how the navigation in the e-commerce environment is created. This form of participation is less direct than in a MUD or Moo and it might even be not transparent to the user at all, so she might not realize being a part of shaping the shop. A "users currently online" function shows one's presence in a system, so it is a kind of social navigation, although there is little opportunity to bring in one's context. Unless—as many systems do—the user is allowed to create an avatar or a description of herself, a popular technique in forums and SNS especially among children and young people (cf. [Boyd, 2007]).

Space and Place

In social software and social navigation, many researchers distinguished between the terms *space* and *place*. According to Harrison and Dourish [Harrison and Dourish, 1996] and Dourish [Dourish, 2006a], space refers to the spatial model of a virtual surrounding. Space is very important, e. g. in *virtual reality*, to allow users to navigate in the virtual space. Place, on the other hand, referred to the experienced environment, the social environment. In Harrison and Dourish's words the difference between space and place is the following:

> Space is the opportunity, and place is the understood reality [Harrison and Dourish, 1996, 67].

One of the authors reconsidered the paper after ten years (cf. [Dourish, 2006b]), adding that perception of space is also socially influenced, but the key difference stays the same. A place has to be in some space. Space and place can be mixed up; sometimes they

[8] http://www.moodle.org

are not easy to distinguish. A focus in social media research is to make a virtual space a place. In our section 3.8.5 on constructionist learning environments, we found the notion of the third place that is a place, but rather related to an informal setting and with the priority for the people to go there to socialize. A place can also be found in a virtual workplace setting. It is more relevant that the users feel it is an environment they like and they connect to. To say it in different words, it should be familiar.

How can such a place be designed in opposite to a space? We saw MUDs and MOOs that allow the users to bring in their context, by creating rooms they find interesting or they know from the real-world. Another example Dourish referred to [Dourish, 2001b, 147] is a media space called RAVE developed at Xerox PARC. In RAVE, researchers worked on a video conferencing system and rearranged the cameras in different ways. Instead of only showing the user in front of her computer screen on the video, they came up with pointing the cameras to her office door and thereby allowing the remote person to get an impression of the real office's context, e. g. colleagues passing by the room or entering the office.

We concluded in our concept in chapter 4 that creating (third) places should be part of creating environments aiming at putting constructionism into context. The MUDs and MOOs offered a great way to create places, as the users could create everything and therefore make rooms with objects meaningful to them and perhaps to others. In other virtual environments these creations are not possible—as least not to the extent one can find within MUDs and MOOs. Social navigation is a possibility to take part in creating places. We will introduce another novel technology which we consider as a way to create places and bring in individual as well as shared context, namely tagging. In our implementation (see chapter 6), we use tagging to let the users create places.

5.1.3 Tagging

We already mentioned tagging at different occasions, giving a very brief definition of what tagging is in the introduction. In tagging systems, users can add metadata or keywords (tags) to different kinds of items (cf. [Marlow et al., 2006], [Golder and Huberman, 2005]), for example bookmarks in Delicious[9], or pictures in Flickr, or songs as in the Internet radio Last.FM[10]. These systems can be huge in terms of user numbers and items they organize, as in the examples above. In that case, we can also use the notion of *social tagging*. On the other hand, tagging can be used in small systems, e. g. personal blogs, or to organize one's emails as in Google's email service Gmail[11].

As a rough definition of tagging, we can again say that tagging is the process of labeling items, e. g. pictures, mails or URLs, with one or more free text keywords. Because tagging plays an important role in this thesis, we will introduce the basic concepts in more detail by giving an example of a tagging process, and introduce some concepts we find in many tagging systems, as we go along. The main and crucial differences of e. g.

[9] http://www.delicious.com
[10] http://www.lastfm.de
[11] http://www.gmail.com

the content-based recommender systems we mentioned before (where a shop employee puts new items to the online shop and labels them with a category) compared to tagging are the following.

1. The labels are not fixed, predefined categories, they can be every text-based description the user can imagine.

2. The user (e. g. of the shop) enters the metadata to existing or user-generated content opposed to the shop maintainer or librarian or any other kind of trained professional.

3. Tags can be used to navigate in a websites or another piece of software.

Basics of Tagging

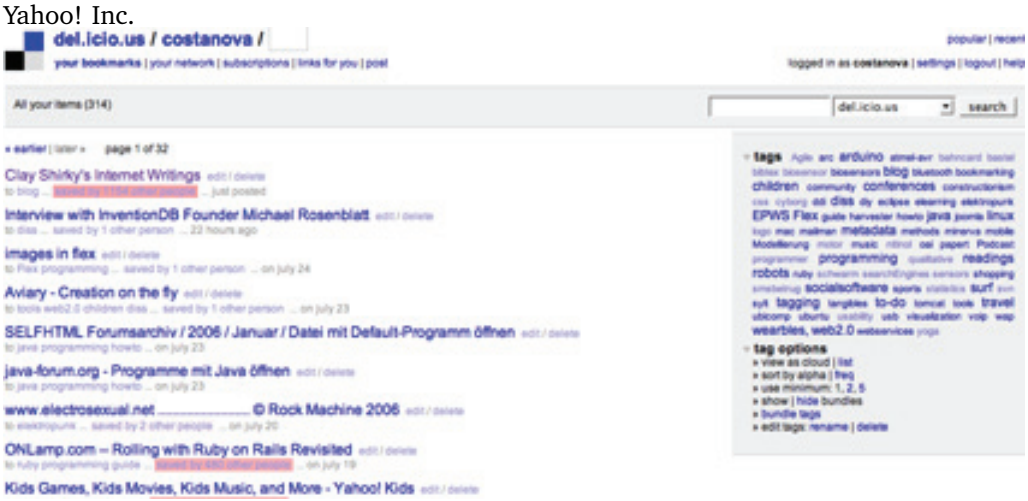
Let us go through the tagging process in Delicious by an example from 3 August 2007. The user Costanova finds an interesting website, for example say Shirky's blog[12]. To find the same website later on again, Costanova uses the browser's tag button to save the URL with Delicious. A pop-up window opens up, suggesting different tags the tagger used before to annotate similar URLs. In our case, these tags are "blog", "community" and so on (see figure 5.1, for more details). As a second way of suggestions, Delicious shows tags other users applied to the URL "http://www.shirky.com/", e. g. "blog',' or "internet". Tags both Costanova and other taggers used (e. g. "blog") are highlighted in a different color. Costanova now can choose which tags to apply or invent new ones. She thinks about how she will most likely find the URL again, and chooses "blog" as a tag. The new bookmark will now be saved and Costanova's personal *tagcloud* (cf. figure 5.2)

Figure 5.1: Suggestions from Delicious. The recommended tags include the user's own tags and popular tags for the URL. Tags the user used before are displayed in green. Reproduced with permission of Yahoo! Inc.®2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo are trademarks of Yahoo! Inc.

[12] http://www.shirky.com/

will be altered. The personal tagcloud is displayed on the right within Delicious. Each tag Costanova used, is pictured in a font size determined by the frequency of occurrence of the tag and usually normalized logarithmically (when a large number of tags is visualized) according to Kaser and Lemir [Kaser and Lemire, 2007]. The tagcloud shows tags in their alphabetical order. The entity of tags of all users in a social tagging system builds

Figure 5.2: Personal tagcloud Delicious. On the left, one can see the URLs the user tagged. On the right, her personal tag cloud is displayed. Reproduced with permission of Yahoo! Inc.®2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo are trademarks of Yahoo! Inc.



a navigation structure that is called *folksonomy* [Mathes, 2004]—short for "folks" and "taxonomy", because of its quality as a bottom-up organized, decentralized structure. In Delicious, the folksonomy is depicted as a (public) tagcloud including either the most popular or the most recent tags as one can see in figure 5.3. We see a folksonomy as an example of social navigation: The user navigates somewhere, because other people tagged the resource in a certain way. Note that Delicious' folksonomy, in contrast to the personal tagcloud, includes only a subset of all tags in the system.

Special forms of Tagging

We said that tags are free text metadata key words that can be added to different, usually web-based, items. However, there are also some special forms of tagging where, instead of free text, defined text is used. Examples are:

1. *Geotagging:* In *geotagging*, people add coordinates to items, e. g. pictures, to specify the location where the picture was taken. Usually a special tool is used for this purpose. Geotagging is quite popular, because it allows mesh-ups—web-based applications that mix data and functionality of several webservices—for example with Google's map service Googlemaps[13]; see the geotagging group in Flickr[14] for further details. An example of such a mesh-up could be an application that shows photos of users with similar interests on the map whenever one points to a city.

[13] http://googlemaps.com
[14] http://www.flickr.com/groups/geotagging

Figure 5.3: Folksonomy Delicious. In the folksonomy of the whole Delicious system the most popular tags are displayed. The tags Costanova has in common with the popular tags are colored red. Reproduced with permission of Yahoo! Inc.®2008 by Yahoo! Inc. YAHOO! and the YAHOO! logo are trademarks of Yahoo! Inc.

2. *Advanced image tagging:* Flickr, for example, allows to tag parts of an image with free text tags (called notes in Flickr). Some social networking sites, e. g. Facebook[15] and StudiVZ[16], took up this idea of image tagging and allow to tag parts of pictures with names of users that exist within the system. Facebook has the same functionality for videos and notes.

We find Facebook's image tagging very interesting, as it might add some user motivations to Marlow's taxonomy (we introduce the taxonomy later, in section 5.1.3). The tagging feature in Facebook not only allows tagging of users, links to their profile will be automatically inserted and messages like "Milena Reichel was tagged in a photo" will appear in the user's profile. We suspect that users also use this kind of tagging feature for awareness and to connect with their friends.

Research on Tagging

In our example of tagging a website as a bookmark in Delicious, we met some fundamental concepts of tagging: tags, tagclouds, tag suggestions and folksonomies. These concepts can be found in most tagging systems. In the example, the user had a certain aim in mind: to ease future retrieval of her data. She was supported in the process of tagging by a number of suggestions the system made. According to Marlow et al.'s taxonomy, tagging systems are different in that respect. The impact of these differences on the users' tagging behavior and the tagging system's vocabulary are not known yet. Academic research on tagging system is quite young, but recently the number of publications discussing related research is growing, as for example the overview article from Marlow et al. [Marlow et al., 2006] showed.

[15] http://www.facebook.com
[16] http://www.studivz.net/

In folksonomies, navigational structures are created from the users' tags [Mathes, 2004] by displaying the most popular tags used in the specific system. Tagging became popular in relation to the term Web 2.0 [O'Reilly, 2005] and is recently a subject of academic research (cf. [Marlow et al., 2006]) although it is not a new technique. The fact that non-expert users instead of trained professionals enhance content with metadata is the important change that makes tagging different and interesting.

There are different research directions associated with tagging: From the information retrieval perspective, researchers try to improve retrieving data in tagging systems by improving the quality of metadata, normalizing and analyzing the sets of tags in a system. Currently, there are even approaches to retrieve semantic information from tags and folksonomies (cf. e. g. [Specia and Motta, 2007]). Other research directions try to identify patterns in the use of specific tags, as the vocabulary of a system evolves over time. Also, visualization of tags is an interesting question, because although tagclouds are the most common visualization, they are not necessarily the best solution for every application. Especially when having huge amounts of tags, usually only the most popular tags will be shown—users with special interests stay out of the tagcloud, because only mainstream tags—typically the most popular or the most recent ones—are included. Another perspective of tagging systems research focuses on statistical analysis of data within the existing tagging systems.

From a social or contextual perspective the question on why and how people use tagging systems is most interesting. Marlow et al. suggested different user motivations on tagging, namely "future retrieval, contribution and sharing, attract attention, play and competition, self presentation, and opinion expression" [Marlow et al., 2006, 35].

In their overview paper, Marlow et al. also differentiated between different aspects in system designs namely "tagging rights, tagging support, aggregation model, object type, source of material, resource connectivity and social connectivity" to classify the various tagging systems that are available on the Internet.

We would like to borrow Marlow et al.'s list of criteria, respectively their taxonomy and explain it in more detail. "Tagging rights" is about who has the right to tag items in the system that can be, e. g. "self-tagging" where a user can only tag items that she created or "free-for-all" tagging where everybody can tag everything. Between the two poles, mixed or hybrid right systems are possible. "Tagging support" refers to help and recommendations the system gives while tagging an item. We have seen Delicious as an example of a successful supportive tagging system. During the tagging process, we could see the tags other users used on the item and tags we used before that might be fitting. In Marlow et al.'s list of criteria that would be "suggestive tagging" opposed to, for example, "blind tagging" where the user cannot see any other tags while tagging. In the "aggregation model", Marlow et al. differentiated between the "bag model" where different users can tag an item with the same (duplicate) tag, and the "set model", where no duplicates are allowed. In the Delicious example, we saw a bag model. Different users tagged Shirky's blog with the tag "blog". In other systems, e. g. Flickr, a user uploads a photo and tags it herself, then sets access rights and maybe allows other users to tag the photo as well. They than cannot use the tags that are already in use for the photo. The

difference between the two models influences usage statistics a lot. The "type of object" refers to the item tagged in the system, e. g. URLs in Delicious. The "source of material" also refers to the items and whether the users create them or tag existing resources, e. g. bookmarks. "Resource connectivity" judges about how the single items are interlinked in the system, for example, by groups. "Social connectivity" is related to "resource connectivity" but refers to connections between users, e. g. by groups, communities and friendships—there are both symmetric and asymmetric connections. In their taxonomy Marlow et al. showed that the criteria related to tagging, e. g. "social connectivity", can have an impact on tagging systems, one could for example build folksonomies based on social connections.

The taxonomy is quite helpful to analyze and classify tagging systems. But still there is little research on the implications of the system's design on the user's behavior. We can, for example, imagine that in a suggestive tagging system fewer new tags will appear compared to a blind tagging system. In the following section, we give an overview over studies that deal in more detail with the question on how people use tagging systems and how the design of tagging systems influences this usage. We assume that the design has a great impact on the behavior.

Golder and Huberman [Golder and Huberman, 2005] identified stable patterns in tagging behavior in the Delicious system. In the beginning, the number of tags for an item is growing and then eventually stabilizes. Golder and Huberman classified tags by the function they have for the user, namely: "identifying what (or who) it is about, identifying what it is, identifying who owns it, refining categories, identifying qualities or characteristics, self reference and task organizing" [Golder and Huberman, 2005]. We can see two different perspectives on the functions of a tag. Marlow et al. considered social functions of tagging while Golder and Huberman deploy a strictly technical view on function.

According to Lerman [Lerman, 2007], users of tagging systems rather use a method she calls *social browsing*. Social browsing refers to navigating through content by looking at what your friends and their friends looked at recently, instead of using tags for navigation. This approach is similar to the "social" motivation Marlow et al. discussed considering their users' incentives.

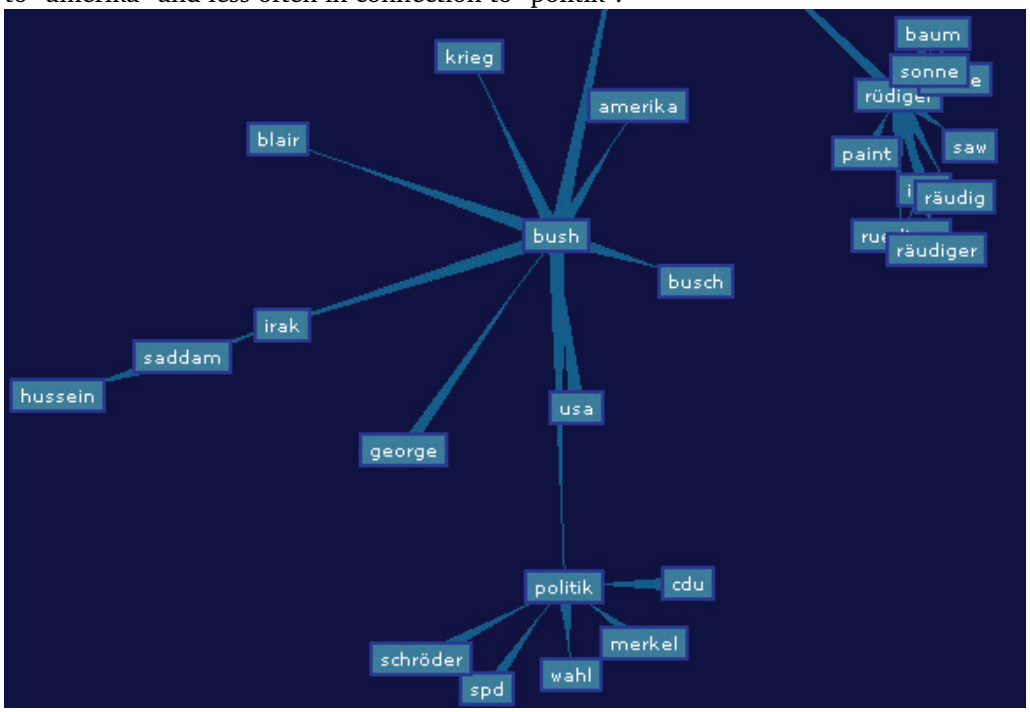
For the organization of a personal tagcloud, users might have different motives to tag than when tagging for the public context. Take for example the tag "seen live" that is quite common in Last.FM. From a personal perspective, "seen live" can help the user to find songs easily. But the tag can also have a different meaning for the user and serve as self presentation in the public context of Last.FM's folksonomy. In the public self presentation the "seen live" tag might indicate that a user saw a lot of bands live and is therefore very involved in a special sub community.

Tag Visualization

We introduced tagclouds as the most common visualization of sets of tags. There is surprisingly little research on optimizing tagclouds or alternative visualizations—at least

according to Kaser and Lemire [Kaser and Lemire, 2007]. According to them, tagclouds themselves could be optimized, because they waste space on screen (through using the different font sizes one creates empty spaces). The alphabetical order that is typical of tagclouds does not visualize any relationship between tags. A technical limitation of tagcloud visualization may be that they are usually web based and rely on HTML output displaying each tag as a hyperlink. Rivadeneira et al. [Rivadeneira et al., 2007] evaluated tagclouds and came to the conclusion that the font size of tags works (as expected) quite well to attract the user's attention but that the layout of the tagcloud should also be considered. Users typically look at the upper left corner of a tag cloud first (cf. [Kaser and Lemire, 2007]).

Figure 5.4: Visualizing tag relations with EMMA. The more similar the tags are the closer they are clustered together. The tag "bush" for example appears quite often in connection to "amerika" and less often in connection to "politik".



Some alternative ideas of tag visualization were made; typically they display correlations between tags and another factor. Russel [Russell, 2006] created Cloudalicious. Cloudalicious displays how tags evolve over time. The number on how often each tag is applied for a certain resource in a given time period is displayed as a graph. Bielenberg and Zacher clustered tags of the Delicious system to foci in their work Groop.us [Bielenberg and Zacher, 2005] using Feld's definition of focus, namely

> a social, psychological, legal or physical entity around which joint activities are organized (e. g. workplaces, voluntary organizations, hangouts, families, etc.). [Feld, 1981, 1016]

They assumed that users with similar interests use tags similarly. The most important tags were displayed in the center of a focus while the related but less important tags were displayed around the center with a smaller font size.

Most of these tag visualizations rely on tags alone and visualize their relations (see figure 5.4 for an example of EMMA—the tag-based recommendation engine we use in Amici). Flickr's taglines[17] is an exception by displaying both tags and items. Of course, having pictures as items is easier than to visualize bookmarks, for example. Clustering tags to groups by their similarity assumes an improved computation of tag relationships which we will discuss in the following section.

Tag Retrieval and Improved Search

According to Golder and Huberman [Golder and Huberman, 2005], the main problems of tags compared to controlled vocabulary are *polysemy*, *synonomy* and *level variation*. Polysemy refers to the problem that a tag can have multiple meanings (see section 5.1.4 for more details on what we mean by "meaning" in this context). Let us take a look at an example. The tag "Apple" might refer to a fruit, but also to a Macintosh computer. Polysemy is the opposite of synonomy. There can be different tags meaning the same thing. To stick with the example, a bookmark about Macintosh computers could be tagged with "Apple", "Mac", "Macs", "Apple_Computers", "AppleComputers" or "Macintosh". While the tags "Apple" and "Mac" have a semantic similarity, "Mac" and "Macs" have the same word root. "Apple_computers" and "AppleComputers" are different notations; the *CamelCase* way of writing—popular among Java programmers—uses capitals to identify a new word beginning in a composed word; the underscore in a different notation does the same. Level variation, according to Golder and Huberman [Golder and Huberman, 2005], refers to how general a tag describes an item. Imagine the children and young people from our vignette tagging a picture of their artifact. They may call it "robot" or maybe "programming". These tags are very general and e. g. for an expert robotics programmer even too general. The expert programmer would be rather interested in a specialized tag, e. g. "LegoMindstorms" or "MindstormsDodgeProgram".

In tagging, we can also find appropriations—as we mentioned before: Some users, for example, use tags like "#mac" to make the tag appear first on the alphabetically-ordered tagcloud.

Spelling mistakes also produce different notations. Single tags can be normalized with methods from language recognition. Sood et al., for example, use a stemmer to identify the morphological root of a tag [Sood et al., 2007]. Because tags—opposed to categories—are not hierarchical and typically users use more than one tag for an item, most items in a tagging system have more than one tag. Therefore, the correlation of tags is a possibility to find similar tags by assuming that two tags have a semantic similarity when they appear together to classify an item. However, the absolute number of appearances of tags is not well suited, because very popular tags have a high probability to appear in combination with any other tag. Imagine for example, a system, where

[17] http://research.yahoo.com/taglines/

programmers tag tutorials for programming languages. A tag like "programming" might appear in combination with nearly every other tag in the system.

Begelman et al. [Begelman et al., 2006] proposed a similarity measure of tags in which they calculated the relative probability for each pair in a set of tags that both tags appear for the same item with a so-called dice similarity measure.

$$Similarity_{A,B} = \frac{2|A \cap B|}{|A| + |B|} \tag{5.1}$$

With this measurement, one has a relative similarity with the advantage that the most common tags do not dominate the results.

Begelman et al. used the similarity measure for a clustering algorithm that allowed dividing a set of tags into several cluster of similar tags. We will go more into detail concerning the algorithm in our own implementation section 6.4.2.

5.1.4 Meaning Creation and Sharing of Meaning in Tagging Systems

Up to now, we discussed concepts related to tagging, showed how tagging systems work, introduced a taxonomy of tagging systems and pointed out current research directions related to tagging. To come back to our initial question regarding context and the sharing of meaning, we can take a closer look on how the users tag and share these tags.

In section 2.2.1, we introduced basic assumptions of phenomenology and their influence on the notion of context we use in this thesis very briefly. When we try to look at meaning creation in social software, and especially tagging, under a phenomenlogogist's perspective, we find some research.

Campbell [Campbell, 2006], who tried to develop a phenomenological framework for tagging based on Husserl, pointed out that tagging allows multiple perspectives and interpretations which, in his opinion, makes tagging so valuable under a phenomenologists' perspective. He considered three aspects as most important.

> The tagging system, therefore, encourages the user to integrate his or her reflection of the resource into evolving structures: the user's own tags, those of other users, and those that are the most popular. [Campbell, 2006, 9]

We think that this point of view fits to Heidegger. For Heidegger, meaning is not something that exists beforehand and can simply be assigned to a phenomenon; it evolves through action.

> Sie [die Interpretation] wirft nicht gleichsam über das nackte Vorhandene eine Bedeutung und beklebt es nicht mit einem Wert, sondern mit dem innerweltlichen Begegnenden als solchem hat es je schon eine im Weltverstehen erschlossene Bewandtnis, die durch die Auslegung herausgelagert wird. [Heidegger, 1967, 150]

In Macquarrie and Robinson's translation, the quote is the following:

> In interpreting, we do not, so to speak, throw a 'signification' over some naked thing which is present-at-hand, we do not stick a value on it; but when

> something within-the-world is encountered as such, the thing in question already has an involvement which is disclosed in our understanding of the world, and this involvement is one which gets laid out by the interpretation. [Heidegger, 1962, 191]

Under a representational perspective, the assignment and sharing of meaning is often described in a manner similar to this: When browsing the web, the user might find an article interesting enough to stop browsing for a moment and to add the article as a bookmark. In the example that demonstrated basics of tagging in section 5.1.3, Costanova came to that point, when stumbling over Clay Shirky's blog. Depending on her context and her interest in the blog, she chose the tags that represent ideas or concepts. While "blog" was a rather broad and general tag (as well as such an idea or concept), she might as well choose something like "dissertation" because at the very moment one of Shirky's ideas could be valuable for Costanova's thesis.

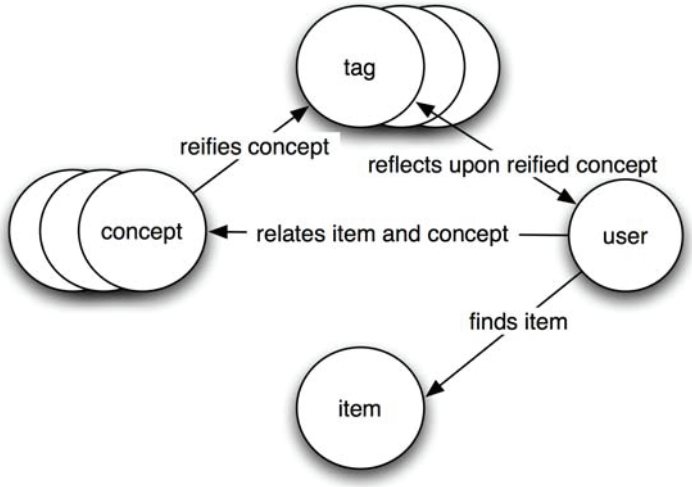
Most tagging systems allow social tagging—they combine tags of all users in a folksonomy—and have a large number of users. That is why one can ask the question, on how the individual's tagging process relates to the creation of a common understanding in a system with many users or in a subgroup. We will take a closer look at the question, as for example Sinha [Sinha, 2006], Marinchev [Marinchev, 2006] or Kohlhase and Reichel [Kohlhase and Reichel, 2008] did.

Opposed to categorizing, the user can relate multiple concepts or ideas to an item through different tags. Marinchev [Marinchev, 2006] claimed that the user will associate different concepts with the item and then eventually build a focus, or—as he called it—a *semantic field* [Marinchev, 2006].

> The tag is a sort of label that references a concept or set of concepts. A cloud of tags is then a collection of labels referring to a cluster of aggregated concepts. [Marinchev, 2006, 36]

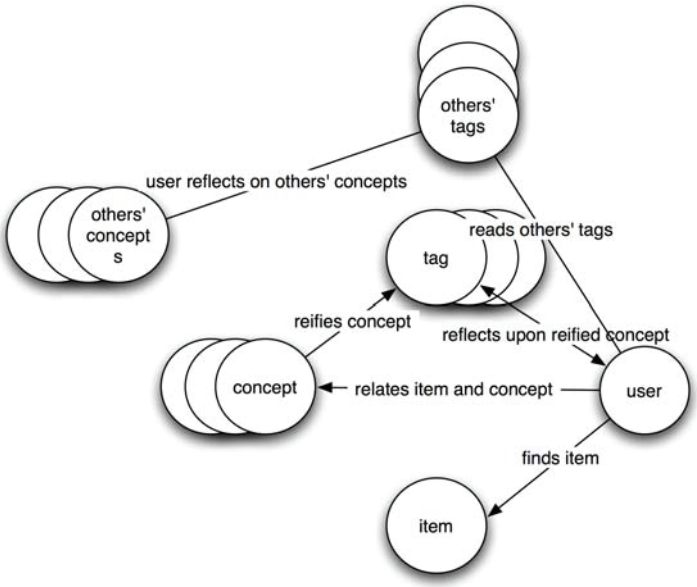
This semantic field is one domain the user connects the tag to. Shirky's blog might be a blog that is generally interesting and Costanova will read it once in a while and at the same time it is relevant for a chapter Costanova is working on right now. Therefore, the tags she chooses are relevant in the ad-hoc situation. When she searches a bookmark in her tagcloud at another time, she might relate the tagged URL to something completely different. She has to interpret her tags in front of a different context. In figure 5.5, we show a model of the meaning creation process of the individual in a representational view. The tagging system might also include other users, and we have seen in the example that a lot of them tagged Shirky's blog, in some cases with very different concepts compared to Costanova, e.g. "economics" or "culture". Costanova gets a glimpse of these concepts, when adding the bookmark to her Delicious account, and possibly changes or adapts her ideas related to the item. At the same time, she is adding her tag to the pool of tags in a system and therefore adds her perspective that might change the other's interpretation of the item. The circle goes on and on. The same process happens when searching for a URL: The individual learns about the others' concepts of the item and then again will probably change their conceptions through adding her tags. When inter-

Figure 5.5: Meaning creation with tags of the individual in a representational view. Figure is inspired by Marinchev [Marinchev, 2006]



preting the tags given by other users one might ask why or who tagged the item with a tag (see figure 5.6).

Figure 5.6: Meaning creation with tags in social tagging in a representational view. Figure is inspired by Marinchev [Marinchev, 2006]



We notice that the main assumption—a tag represents a fixed meaning—is present in that view. From a phenomenologists' perspective, we would like to add that there cannot be a fixed idea a tag represents. When interpreting a tag as a zeug, its meaning will arise out of the interaction. As soon as our user browses the web and finds an interesting website, there is no such thing as the abstract idea of "blog", the zeug blog is connected to the activity of web browsing, the user's intention and of the sentence the user reads

and so on. Sharing of meaning then can only happen, when the users share a lived experience.

We cannot solve the contradiction between phenomenologist's and representational view here and share Campbell's point that tagging, because it allows assigning multiple free text labels to one item, allows a greater flexibility than a distinct categorization. However, in research on tagging systems we find that the representational perspective dominates. Under this perspective, meaning is fixed and a tag represents a certain meaning in a certain domain.

5.1.5 Studies of Existing Tagging Systems

We have seen in our section 5.1.3 on tagging that there is research on how people use tagging systems and that the research supports the claim that tagging allows to integrate one's context on a personal level as well as within a group or community of users. Most tagging systems are based on groups and the group of users creates a folksonomy of the system.

Little is known up to now, on how (and to what extend) children and young people use these systems and whether or not there are differences in their behavior compared to adults' behavior. There are many studies on tagging systems that regard the whole system as one community. See for example Geisler and Burns [Geisler and Burns, 2007], and their study about general tagging strategies in YouTube. The authors called the set of all YouTube users the YouTube community.

Tagging systems allow building sub-communities that may as well be constructionist communities or communities of practice. These sub-communities can have their own tagging strategies and build their own vocabulary. While communities often have a large number of users, finer grained subgroups through e. g. friendships, or shared interests, exist. We also pointed out before that users who use tags in the same way possibly share the same interests, Bielenberg and Zacher [Bielenberg and Zacher, 2005] used that as the main hypothesis for their Groop.us system. We suspect that interest groups or groups of friends use similar tags.

To find out more on (especially young) users of tagging systems and sharing of meaning and interests through tags, we decided to study an existing tag system namely YouTube with the well-known method called *social network analysis* (SNA) that seems suitable to find out more on social connections between users related to tags. Our hypothesis was that in social groups, e. g. a group of friends, a common tag behavior arises (and that friends use the same tags more often than members of a community of interest).

Social Network Analysis

Social network analysis is a method to find patterns and structures in abstract models of (social) groups, for example communities or market systems. It is a huge research field; we do not want to go into SNA in detail, and we only touch it briefly by using it to model the tagging system YouTube. In SNA, groups are modeled as graphs or—as a

special form of graphs—networks. There are variations in the definitions of properties of graphs, networks and corresponding concepts, that is why we define the most important terms according to de Nooy et al. [de Nooy et al., 2005] at this place. We only refer to the concepts that we actually use in our analysis.

1. *Network:* The definition of network in SNA is the same as the one of a simple graph in graph theory. That is: A graph G is a set of nodes V and a set of edges E that connect the nodes. In SNA the graph or network typically represents a social structure.

   G={V,E}

2. *Vertices/Nodes: Vertices* or *nodes* represent the actors of a network. In SNA, a node often represents a person, but it can as well stand for, e. g. a material, an organization or a tag.

3. *Edge:* An *edge* or *tie* is a connection between two nodes. Opposed to an *arch*, an edge is not directed. In SNA, edges represent relations, e. g. “is a relative of”, “is a friend of”, “belongs to the same organization” or “trades with”.

4. *Degree:* The *degree* of a node is the number of edges the node has. Because we work with an undirected network, we do not differentiate between an incoming or outgoing degree. A node that represents a person might be connected to three other persons by the relation “is friend of”. The degree of the node would then be three.

5. *Component:* Because we work with undirected graphs a *component* is a maximal connected sub graph. That means, every node in the network is reachable from every other node and one cannot add a vertice to the graph leaving it connected.

6. *Weighted:* An edge can have a certain *weight*. For example, in a SNA where nodes stand for countries and edges for trade between the countries, the weight of the edge could represent the amount of goods traded between the two countries.

7. *Directed:* An edge that has a direction is called *arch*. It represents relations that are unidirectional, e. g. a country exports to another country, but does not import from it.

8. *Undirected:* An edge means the relation is bidirectional and symmetric. For example, “I am your friend and you are mine”.

9. *Density:* The *density* of a network is the relation of (real) connections in the network to possible connections. A high density in a social network means there are lots of interactions or relations between the actors.

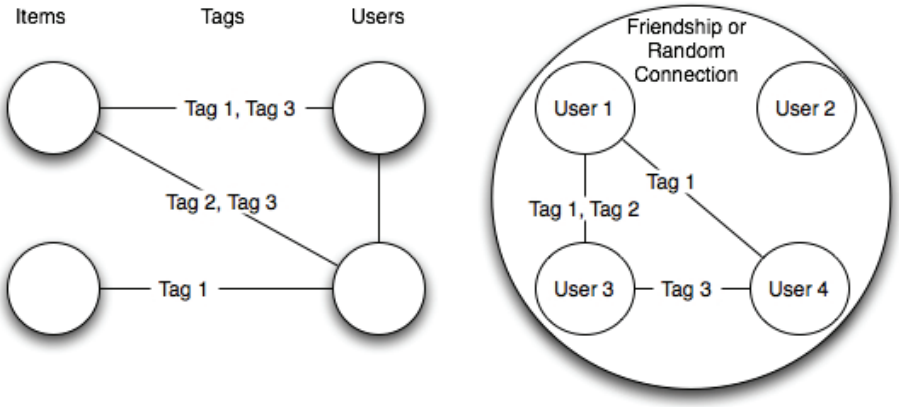
YouTube

YouTube is a well-known service (cf. the JIM study we discussed in section 5.1.1) to upload and share videos. It was founded in 2005 and acquired by Google in 2006.

Videos can be uploaded and require both a tag and an assignment to a given category during the upload process. To borrow Marlow et al.'s criteria again, YouTube is a self-tagging (users tag only their created content), blind tagging (users cannot see tags while tagging), set-model (no duplicate tags for one item are allowed) system; the objects users tag are videos and the content is created (or rather uploaded) by the users, items can be grouped in *channels* or *communities* but no channel or community is suggested by the system during the upload process. These characteristics certainly have an impact on how the users use the system, e. g. we suspect that more misspelled tags appear, because of the blind tagging. Users can be connected by friendship or *family* status. The users can also join channels or communities and get notifications when new content appears in these channels or communities via RSS feeds.

In tagging research, a tagging system is usually modeled as a tripartite network with an interrelation between tags, items and users, so that each user can tag an item with a tag (cf. [Marlow et al., 2006]). That view does not apply to YouTube, because users upload and tag only their own videos. Every video is in the system only once (at least under the same name). Bookmark services on the other hand allow different users to tag the same item multiple times. In the case of social bookmarking, e. g. in the Delicious example, the model of tagging systems as tripartite network works fine. We model the users of YouTube as nodes and add a connection, whenever they used the same tag (see figure 5.7 to compare the different models visually).

Figure 5.7: Tagging systems modeled as tripartite network according to Marlow et al. [Marlow et al., 2006] (on the left) opposed to our model of YouTube (on the right).



The reason to use YouTube for a study is that it provides an *application programming interface* (API). An API allows every developer to write applications that access information within the YouTube system. One can, for example, access user profiles or popular videos. The items shared in YouTube are videos and therefore are visible and understandable, even if they are in different languages. That is why the videos can be used in tests with and children young people (opposed to, for example, bookmarks that require a lot of websites to be read). The JIM study also found out that YouTube is well-known and used among youngsters. The age of the user is a mandatory information when reg-

istering in YouTube (note that YouTube requires a minimal age of thirteen[18]), so we can analyze data in relation to the users' indicated age. Some content is available only for people older than eighteen (that might explain the peak in the age curve at the age of nineteen that one can see in figure 5.8). Young people might join YouTube entering an age that allows them to view all content. Age as well as all other information of the users is not perfectly reliable, users can enter whatever data they want to enter. We still use the data of YouTube and suspect that a lot of users enter "real" data. Still, one has to be careful with the results considering wrong user details. In the data we harvested, we found several data sets where the user used the description in her profile to make clear that she is actually younger than thirteen, but she had to enter the wrong age to be allowed in the system. An example is the user LithuanianGurlLexy, who indicated an age of thirteen while her "about me" text is:

> Birthdate:August 4,1996
> Age:11
> Sign:Leo
> ["About me", user profile LithuanianGurlLexy]

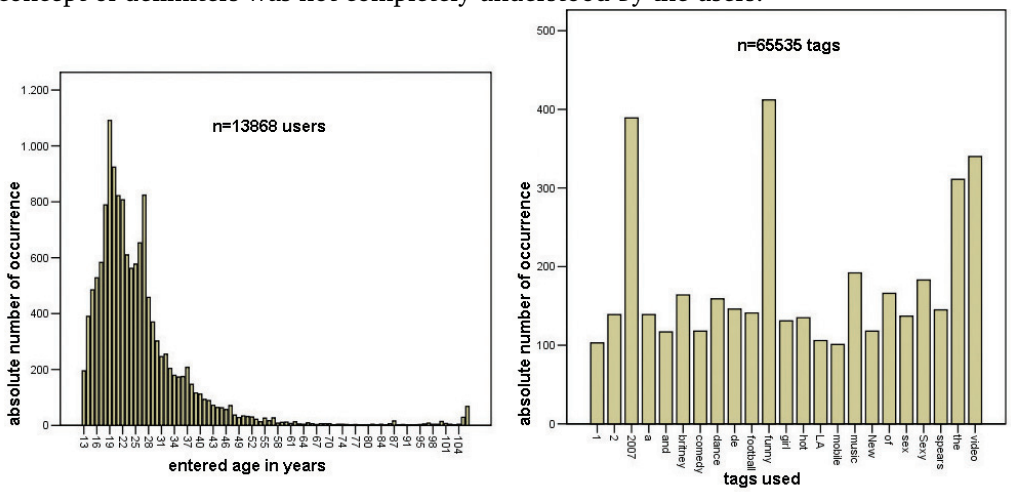
To gain a general impression of contents, users and tags in the YouTube system, we started by analyzing popular content.

The data sample we took consisted of the featured, the most popular and the most discussed videos during 22 August 2007 and 27 September 2007. It was acquired by using YouTube's API and by parsing YouTube's RSS-feeds of the most popular and most discussed videos hourly during the time period. We harvested videos and user profiles distinct in their authors. The sample included 13868 users and as well as 13868 videos with a total amount of 65535 tags (28364 distinct tags). The average indicated age (calculated as mean) of the users in the sample was 23.11; 68.2 % of the users indicated they were male. Figure 5.8 shows the age curve of our sample. Of the distinct tags, 7289 were used only once. The most common tags in the data sample were: "2007" (with 0.6 %), "funny", "video", "the" and "music" (see figure 5.8). The usage of the tag "the" ("a" also occurs quite often with 0.1 %) suggests that the users do not fully understand how to use YouTube's tagging system, e.g. that blanks are interpreted as a divider—or *delimiter*—for new tags. When we compare the popular tags for example with Delicious' folksonomy (that we showed in our example of tagging in section 5.1.3), we see that these obviously wrong tags do not appear in Delicious and that Delicious' folksonomy centers around certain Internet related tags (e. g. "javascript") while YouTube is rather fun related.

A main difference between young people (or, more precise, users that entered an age below nineteen in their user profile) using YouTube compared to older ones, is the way they use the system as a SNS and connect to each other with YouTube's friendship functionality. Young people have significantly more friends than the older users (cf. figure 5.9). That behavior is the main difference, there are no significant differences in how many videos younger people watch or download compared to the older users

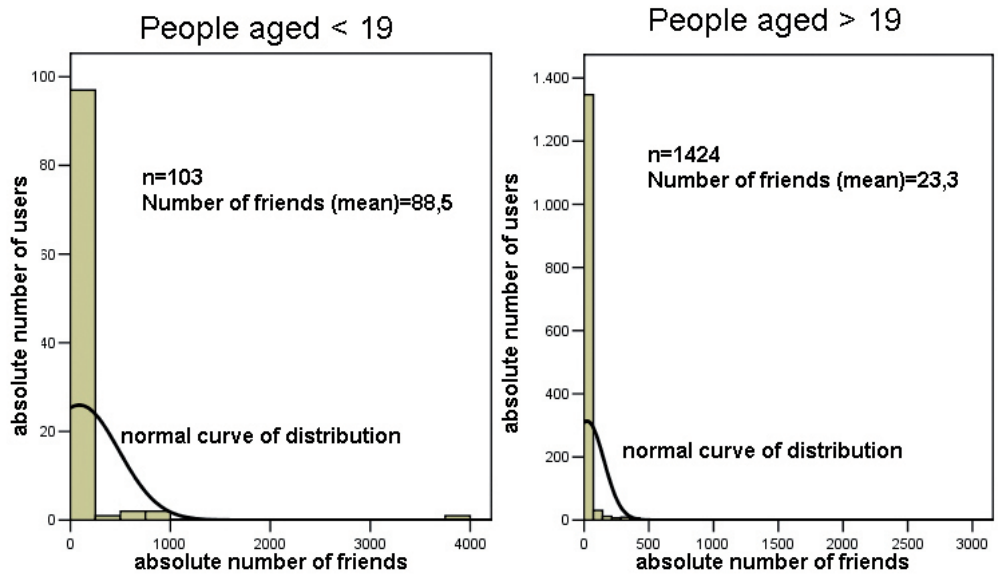
[18] http://www.youtube.com/t/terms

Figure 5.8: Basic data YouTube popular videos. On the left one can see the users' indicated ages and on the right the most popular tags. A lot of users are in their early twenties. The most popular tags in the system include tags like "a" or "of"—possibly the concept of delimiters was not completely understood by the users.



of the system. The main tags of YouTube do not suggest that people use the system

Figure 5.9: YouTube friendships depending on indicated age. Young people (aged under nineteen) have 103 friends in average, while user aged from twenty years on have 23.3 friends in average.

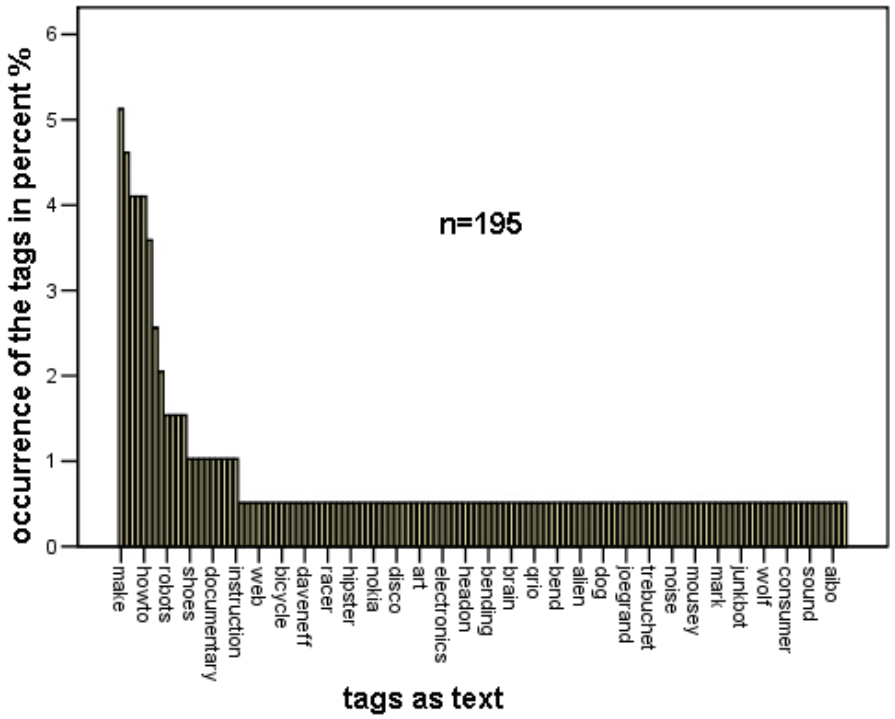


as a learning environment or as constructionist community. "Fun" in whatever ways seems to be the most important topic. We claim, that in spite of this first impression, a system like YouTube offers spaces for special interest communities, as e. g. learning communities. The smaller communities can take these spaces and design them according to their interests and requirements. We tried to find a constructionist community in

the system and identified the Make community for this purpose. The Make community is actually called “Make magazine” and belongs to the US-American magazine Make[19] which publishes how-tos and do-it-yourself projects mainly for IT and technology projects (for example how to hack a hoover robot). For a detailed analysis of a constructionist community, we harvested all users of the community.

On 21 October 2007 the Make community had 1667 members. The users indicate a greater age (33.81 in average) and the percentage of indicated males was slightly greater (74.5 %) than of the average users in the YouTube system. Still there are young people in the Make community (6.75 % are under nineteen years old). The most common tags in the Make community are different from YouTube’s popular tags. Most common are: “make” (5.1 %), “diy”, “bre”, “instructions”, “howto” and “pettis” (cf. figure 5.10). We notice there are less obviously wrong tags comperated to YouTube’s general tags, e. g. “the” or “a”, and the tags clearly center around making things, as one would expect from a constructionist community. There are tags within the Make community that are

Figure 5.10: Most common tags in the Make community. Tags that indicate an interest for the main topics of the community, e. g. “make” or “howto” preponderate.



clearly understandable for someone from outside of the community, e. g. “make” and “diy”. “Pettis” and “bre”, on the opposite, do not make sense for an outsider straight from the beginning. But within the community, they are well understood or at least used often. With the context knowledge that “Bre Pettis” is the producer of a weekly video

[19] www.makezine.com

show for the Make magazine, the tag is understandable and makes sense in the Make community.

Within the Make community, we identified users that are not only members of the community but connected in a stronger way by using the friendship functionality of YouTube. The users that do not upload content themselves typically do not have friendships, either. We suggest that they use YouTube as a passive medium. Still, they are different from the passive users the JIM study reported about, because they are registered YouTube users opposed to anonymous users that only watch videos on YouTube passively.

From all active members (meaning they use the friendship function) we harvested the data of all their friends, leading to a total amount of 6397 users related to the Make community.

The first step of our following analysis was to identify a group within these users that was related through friendships densely. To identify such a strongly connected group, we processed the data in the following way:

1. First, we performed a network reduction and removed vertices with a degree less than one. In our model these vertices represent users that have no friendships.

2. Afterwards, we removed all multiple lines and loops. Multiple lines can evolve through unsymmetrical friendships, e. g. I send you a friend request that you did not accept yet.

3. Within the resulting partition we searched for components.

4. We extracted a component.

The component represented a group of users that were strongly connected by friendship. Our question was, whether friends share the same interests and use tags similarly—we wanted to find out if there is a correlation between friendships and tag usage.

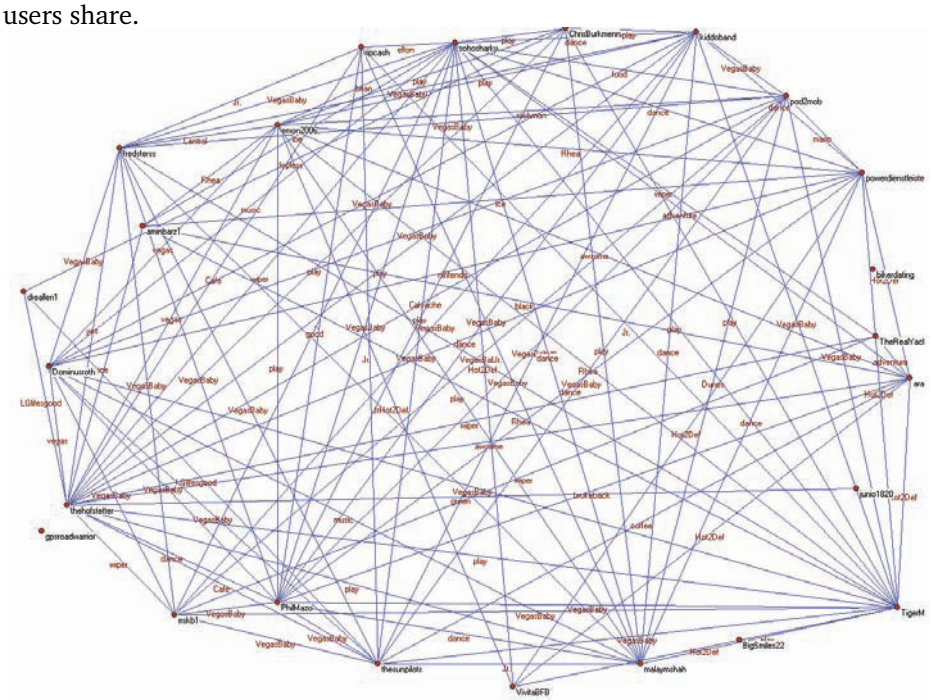
If one models users as vertices and tags as connections between users, one can interpret the tag-users relation as social network. We harvested all tags of the strongly connected users—the component we just extracted—and added an edge between a pair of users if they shared a tag, meaning they used the same tag. The edges therefore represent “shared” tags in our model (cf. figure 5.7).

1. We removed all multiple lines and loops. A loop could evolve when a user used the same tag twice and a multiple line could evolve when the pair of users both used the same tag more often.

As a result, we had twenty-four users who are connected through their tag usage. You can see from the resulting graph (see figure 5.11) that these users are mainly connected with each other. There are only two users, “BigSmiles22” and “bikerdating”, that belong to the component but are a kind of outsiders, as they are not connected to other users in the graph—they belong to the group of friends, but they do not use similar tags. The density of the network without loops was 0.4057971.

In the next step, we created a network for comparison without the friendship connection. We chose a random set of the Makers and their friends and reduced the network by deleting loops and multiple lines. Then we added an edge, whenever a pair of users used the same tag. There might be friendships as well within the set of randomly picked users, but the density of the network is still significantly less than the one of the friends—namely = 0.2554113 (see figure 5.12 for illustration). We were astonished that the identified networks shared the same tags, but that the tags were mostly not connected to making things. In both networks popular tags were: “VegasBaby”, “Canto”, “Hot2def” and “play”.

Figure 5.11: Connection through friendships and tag usages. The vertices represent users of the YouTube system connected to Make that are “friends”. The edges represent tags the users share.



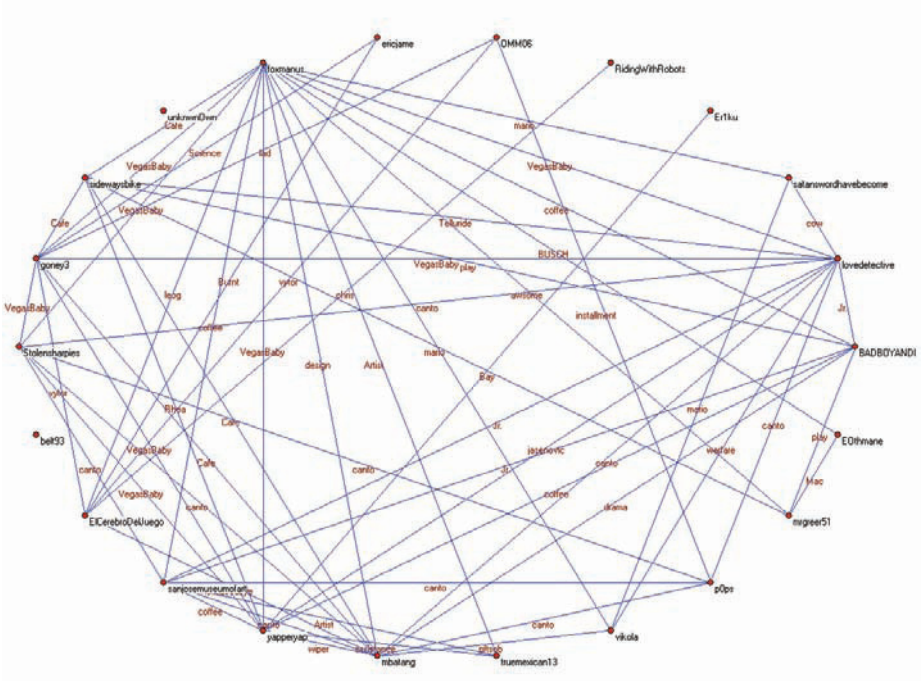
Results of the Study about YouTube

More than half of the selected users are connected through tags and friendships at the same time, they are friends and they share at least one tag. So there seems to be a connection between friendships, interests and the tags that members of YouTube’s Make community use.

We can point out that YouTube is a system that is also used by young people—if we assume the user’s age is correct for the majority of them. The children and young people’s usage behavior differs from the adults’ behavior, e. g. young people tend to use YouTube as a social networking site.

There are constructionist learning communities within YouTube (or at least we see the Make community as one, because they discuss and exchange their own constructions)

Figure 5.12: Connection through tags in a set of random users. The vertices represent random users of the YouTube system and Make. The edges represent tags the users share.



and the tags they use seem to express their common interest. To refer to Schütz' notion of provinces of meaning: YouTube offers places for different provinces of meaning. Some (most) users use it in the leisure time world—for fun only. While others—the makers—use it in their world of construction activities (that of course can also be part of the leisure world).

Sub-communities connected by friendship are more likely to use the same tags than random users within the community, but although the friends are related to a certain community their main interest does not center on the tags of the larger community. As we assumed, there is a common vocabulary in the interest group, but also in the friendships. We do not know how, the individual interprets the tag within the community—if there is a common understanding of the tags—but we assume that such an understanding builds up through watching videos, reading the tags and then uploading one's own videos.

5.1.6 Conclusion

Within this section, we showed that social software allows users to connect and to ease communicating with each other, not only in a one-to-one but also in a one-to-many or even many-to-many relationship. Children and young people are very familiar with using the Internet for that purpose—in general for communication and sociability—as well as with using novel technologies, e. g. the technology of tagging . Social software seems to be connected to the world of friendships and is part of the world of everyday life of children and young people. We introduced tagging and related concepts along with current research topics in the field.

We also showed approaches of social software that try to consider the user's context by, for example, creating places and letting the user take part in shaping her environment through using it (being the case with social navigation and some recommendation systems—namely recommendation systems that rely on social filtering). By using the concept of places, designers try to consider social context in virtual spaces, because the users seem to prefer virtual environments they experience as social reality.

We discussed social navigation as one concept that involves the user in creating and designing navigation in websites.

With the technology of tagging, we introduced another concept that involves the user strongly—a novel technology that allows the user to integrate her individual context in the form of tags. When discussing research and theories on how creation and sharing of meaning happens in tagging systems, we were confronted with a different view on meaning (opposed to the *hermeneutic* notion we discussed before). In this representational view, concepts or associations are reificated in keywords or tags. These individual tags are (at least when we consider social tagging) in a close connection to other people's tags especially in suggestive tagging systems and people can share the meaning. From the hermeneutic perspective, meaning is not fixed in the user's mind and therefore can't be reified in a tag. That is why we have a contradiction between two different approaches that we cannot resolve.

However, both perspectives stress the importance of allowing multiple interpretations in designing computational artifacts. Tagging is more flexible than a rigid category structure and allows adding more than one tag and the tags are user-generated. That is why we consider tagging as a good technology to design for flexibility and to allow multiple interpretations. It is a technology we can use to implement our design principle nine: Design for wide walls and for flexibility.

Within a more detailed analysis of the video sharing service YouTube, we have shown that this focus on the social web is more common amongst children and young people than amongst adults. The children and young people use YouTube as SNS—we can call that usage pattern an appropriation. The YouTube system is flexible enough to be used in the way the young people do, but also the adults can use it the way they want to use it. Tagging in the YouTube system was also used in different ways—misunderstood (we could as well consider the practice as appropriation), but also in a very specific community vocabulary. Tagging is a very flexible technology, because it uses free text as a medium. The only limitation is the length of the words to express everything one can say with text. However, in the case of YouTube, we saw that working with the delimiter is a concept that is not grasped by all users.

We were not sure whether tagging was a suitable technology for children and young people, because studies on search behavior of this special target audience showed that the young users do not use keywords for searching (cf. [Bilal, 2000]). Within the YouTube system and backed up by the results of the JIM studies, we showed that the children and young people are not only able to, but do use tagging systems. Still, there might be huge differences in how they use these systems depending on the age of the children and young people, for example.

As a conclusion, we can say that social software can support interaction between users, it is therefore a suitable technology to ease community building. With social navigation, especially with tagging, we introduced an approach that allows the users to design their own place by bringing in their context. Tagging is a flexible technology and can be used to create an own .vocabulary. When we look back to our concept, these are the characteristics we considered important. Therefore, we conclude that tagging is a technology we can take up for designing a constructionist learning environment that considers the users' context.

5.2 Smart Textiles

I speak through my clothes. [Eco, 1973]

Clothes are good at speaking. In Crawford's definition of interactivity (that we discussed in section 2.1 in more detail), he differentiated between thinking, speaking and listening as essential parts of an interaction. The traditional role of clothes is to speak. The wearer communicates something, e. g. her social status, by wearing a special set of clothes. The ability of clothes to listen is traditionally less important. Still, clothes can listen through reacting to the wearer's movement, for example, lifting a hem line when the wearer is dancing. Thinking has no place in traditional clothing and fashion.

In this chapter, we will give an overview over the field of smart textiles with the focus on applications in fashion and interaction design. How about smart textiles and interactivity? Does the "smartness" of smart textiles improve the textile's ability to listen and think? And how do smart textiles relate to the context they are worn in? Is this relation different from traditional clothes?

The first question that comes into mind about smart textiles is: What exactly are smart textiles? The term smart textiles summarizes a wide range of materials that are soft and feel like traditional textiles, but can sense and react to their environment. Conductive threads can be considered as smart textiles—the thread itself can only conduct electric signals, but in combination with sensors or controllers, the textile product can react to its environment. For a more precise definition, we would like to refer to Kirstein et al. [Kirstein et al., 2005] who defined *textile* as:

> Materials are considered to be a textile when they consist of drapeable structures that can be processed on textile machinery. [Kirstein et al., 2005]

On the other hand, they define *electronic*—they talked about *electronic textiles* instead of smart textiles— the following way:

> "Electronic" means that a system is able to exchange and process information. [Kirstein et al., 2005]

Smart or electronic textiles are drapeable structures that are able to process information, if we put together both parts. "Pure" smart textile applications (meaning they exclusively use drapeable structures) that are interactive rarely exist up to now, usually a lot of traditional electronic components, e. g. power or control units, are combined with smart textiles.

Related research fields (or, in some cases, other names for the same field) are computational garments, soft computation, textile-based computing, smart fashion and electronic textiles.

Research on smart textiles was strongly related to *wearable computing* in the beginning and a lot of research was conducted in the military domain (cf. [Kirstein et al., 2005], [Berzowska, 2005], [Lee et al., 2005]).

Mann from the MIT—one of the pioneers in wearable computing—coined the term wearable computing as:

> A wearable computer is a computer that is subsumed into the personal space of the user, controlled by the user, and has both operational and interactional constancy i.e. is always on and always accessible. Most notably, it is a device that is always with the user, and into which the user can always enter commands and execute a set of such entered commands, and in which the user can do so while walking around or doing other activities. [Mann, 1998]

According to Lee et al. [Lee et al., 2005] the placement of smart textiles in wearable computing and therefore in technology institutes and military, led to cyborg aesthetics and only to a weak connection to fashion design. On the other hand, the first artistic and design oriented applications, as well as criticism on wearable computing, also came from MIT scientists. Picard, director of the Affective Computing Research Group at the MIT[20], defined the term *wearables* quite differently from Mann:

> Wearables are computational devices that are worn as an article of clothing or jewelry. [Picard, 1997, 227]

Post et al. wrote in one of the key publications of the new approach to smart textiles:

> As a result, most wearable computing equipment is not truly wearable except in the sense that it fits into a pocket or straps onto the body. What is needed is a way to integrate technology directly into textiles and clothing. Furthermore, textile-based computing is not limited to applications in wearable computing; in fact, it is broadly applicable to ubiquitous computing, allowing the integration of active elements into furniture and decor in general. [Post et al., 2000, 840]

Post et al., especially Orth, who later on founded the company International Fashion Machines[21], developed applications that combined technological research on smart textiles with art and interaction design. The applications included a jacket that acted as a musical instrument with an embroidered keyboard or the Firefly dress that emitted ever changing light patterns. As Post et al. pointed out, the applications are not limited to clothes, and therefore they also created, for example, an electronic tablecloth. This research was still done at a pure technology institute around the year 2000.

Afterwards, design of smart textiles slowly started to infiltrate design institutes and companies. Berzowska, who is also an MIT graduate and a part of the International Fashion Machines team, founded the XSLAB[22] placed in the Faculty of Fine Arts at Concordia University in Canada, as the first interdisciplinary institute for smart textiles. The research lab is strong in technology research as well as in interaction design; we will discuss some projects made at the XSLAB later in this section. In Europe, two graduates from the Interaction Design Institute Ivrea[23], Rosella and Genz, founded the company Cute Circuit[24] that is well known in the world, especially for their product Hug Shirt that

[20] http://affect.media.mit.edu/
[21] http://ifmachines.com/index.html
[22] http://www.xslabs.net/
[23] http://interactionivrea.org/
[24] http://www.cutecircuit.com/

we also describe in more detail later on. Following these pioneers, many art, design and textile schools started to research and teach in the smart textiles area.

Though applications can combine the new technologies with art and design aspects, still the most important question remains: What to do with smart textiles? Do we need and want smart textiles? On the market, we can find different application fields and some (but only very few) commercially available products. Application fields—according to the smart textile research company Ohmatex [Ohmatex, 2007]—include sports and medical wear, work wear (e. g. communication features in worker's clothes or safety features) and accessories (most common are MP3 player related accessories, e. g. speaker, or control elements). We give examples in a later section. New markets could be, for example, shielding in medical applications, but also in automotives or buildings.

5.2.1 Classifications of Smart Textiles

To get an impression what smart textiles are, we give a brief summary of smart textiles classifications by material characteristics and textile working techniques. These classifications are mainly from a textile perspective, but integrate a technical view through asking how the textiles react to their environment.

From a material point of view, we can differentiate between smart textiles that serve as actuators and sensors with an *integrated circuit* (IC) to do the processing and the ones without. Smart textiles that still rely on ICs are called *active smart textiles* according to the project TeTRInno SmarTex [TeTRInno, 2007].

One step ahead are *ultra smart textiles* that process data from the environment without an additional IC. These textiles are also called the *third generation* of smart textiles. You can differentiate between smart textiles by their generation. There are different definitions for each generation, we will use the following one here:

1. *First generation smart textiles:* traditional natural fibers, processed with traditional techniques, passive textiles.
2. *Second generation smart textiles:* alternatives to substitute natural fibers, e. g. synthetics, active smart textiles.
3. *Third generation smart textiles:* current high-tech materials, ultra smart textiles.

5.2.2 Materials

For most prototypes, in interaction design and in the EduWear project, we mainly work with active smart textiles due to their relatively good availability and the possibility to process them in textile machinery.

For active smart textiles, a lot of different materials are used. We will not introduce all groups, because from an interaction point of view, they are not all relevant. In other fashion sectors, e. g. sportswear and functional wear, materials that allow thermo regulation and breathability through developing special membranes play a more important role. We concentrate on sensors, actuators, connections between the parts and ICs as "intelligence" and give a short introduction of the materials.

Shape memory materials. These materials are used as an actuator in smart textiles. Although most *shape memory materials* are not textile, they are often used and researched on in the smart textile area. Shape memory materials is an umbrella term for

> a set of materials that, due to external stimulus can change their shape from some temporary deformed shape to a previously 'programmed' shape. [Honkala, 2006, 85]

An example of shape memory materials are shape memory alloys, e. g. nickel titanium (better known under the trade name Nitinol—NiTi). Shape memory alloys can also be based on polymers. The materials react to a temperature stimulus. They remember geometry—a shape they are formed to—and have two temperature phases. One—the *austenite*—to remember a shape (also called the training of the material) and one—the *martensite*—where the alloy transforms back into the remembered shape. Another characteristic of the materials is the *superelasticity* when they are in their martensite phase (cf. [Honkala, 2006]). With shape memory materials smart textiles can change their form depending on stimuli from the environment. This is what makes them interesting for many researchers. Shape memory alloys are quite a new material and controlling them with ICs and resistive heating is currently in research. It is easy to overheat the material and burn textiles. That is why there are only few products on the market.

Chromic materials. Chromic materials change their color after being exposed to a certain stimulus. In case of thermochromic material, the color-change starts when the material is heated up, in photochromic material UV-light is the stimulus.

Luminescent materials. *Luminescent* materials emit light, just as for example LEDs do. There is a lot of research on these materials, but most prototypes we know of still use electronic components for light emitting. *Electroluminescent* materials emit light when current is passed through. Some materials are available, e. g. electroluminescent foil, or electroluminescent wire which is often used for interior design of automotives or in backlights for displays. We use electroluminescent wire for the Smart Dance clothes that we will introduce in a later section. Phillips' research project Lumalive[25] is well known for engineering new luminescent materials—although Lumalive uses a LED display technically and no luminescent smart textile materials (cf. figure 5.13).

Conductive materials. These materials are commonly used as connections and also as sensors. The simplest example of using conductive (textile) materials as sensors is a textile switch, first described by Post et al. [Post et al., 2000]. They placed two layers of conductive yarn or fabric on top of each other, kept apart by a layer of non-conductive material. Once pressed, both layers touch each other and close the circuit.

Most conductive yarns and fabrics consist of a mix between non-conductive and conductive particles, some yarns on the market are non-conductive yarns with steel (or another metal) coating or metal (or polymer) fibers spun with other fibers. Conductive

[25] http://www.lumalive.com

Figure 5.13: A couch with Lumalive by Phillips. Picture from Phillips http://www.research.philips.com/newscenter/pictures/downloads/. Free to use when referring to the source.

fabrics exist, for example weaves, knits or non-woven materials. The materials are washable, even with traditional components attached, according to Buechley and Eisenberg [Buechley and Eisenberg, 2007], but might loose their conductivity through the process of washing them, according to Tao [Tao, 2005].

IC and circuit boards. ICs are obviously not textile, but a necessary part of active smart textiles. Current chips are mostly made of silicon; they are hard and not flexible. The ICs usually sit on a circuit board and some alternatives to *printed circuit boards* (PCBs) exist. It is possible, to make circuit boards using conductive yarns or fabrics combined with different textile techniques. Post et al. worked with embroidery in commercial embroidery machines, using conductive yarn directly in the bobbin of the embroidery machine to create a textile PCB [Post et al., 2000]. Buechley [Buechley, 2006] cut out pieces of conductive fabric with a laser cutter and ironed them onto (non-conductive) fabric. An alternative for smart textiles are PCBs made of polymers (cf. [Tao, 2005], [Post et al., 2000]) that can have a flexible quality. Printing is also explored as a technique to make textile PCBs. We call PCBs where the entire board is made out of flexible (textile) material in the following part of our thesis *textile PCBs*. The electronic components, for example the IC, are still hard and electronic. An advantage of textile PCBs is the possibility to integrate them into clothing without feeling them, because they are

soft. The possibility to design textile PCBs aesthetically, as one can do with embroidery, is also an advantage.

5.2.3 Techniques

Usually textiles are processed by various techniques using advanced machinery. Conductive threads can be used with traditional textile machines with restrictions, because of the stiffness of the material, and because the metallic material can cause damage to the machines. The textile techniques have a huge impact on the design of applications, because they restrict layouts through their structure and influence material characteristics, e. g. conductivity. In the following part, we introduce some textile techniques and their acceptability to use them for smart textile processing.

Sewing and embroidery. In embroidery, one stitches designs into a fabric. Machine embroidery can be done either manually with a standard sewing machine or by a fully programmable embroidery machine. Industrial embroidery machines can have multiple heads that allow using a number of different threads in the embroidery process. These machines can produce precise patterns of different kinds of designs with advanced software. Most machines can work with conductive yarn, at least in the lower bobbin. Embroidery is often used to create circuits in smart textiles due to the little restrictions in layouts (cf. [Post et al., 2000], [Buechley, 2006]).

Weaving. There are different forms of woven structures, but generally fibers can be woven into two directions: the warp and the weft (cf. figure 5.14). Due to this structure the possibility of weaving complex circuits is limited, the conductive threads in warp and weft will necessarily cross and cause electric connections. The bending angle of the fibers is also important, a small angle can damage the textile structure and conductivity (cf. [El-Sherif, 2005]).

Knitting. In knitting, yarns are connected through loops. This structure bends the yarn extremely, but allows more flexible patterns than a weave. Knits are looser and more flexible than weaves that can have an impact on the electronic characteristics of the material (that is why we use knits as flex sensors in the EduWear project).

Printing. Conductive inks can be used for printing very flexible and precise patterns [Meoli and May-Plumlee, 2002]. Especially with digital printing, all kinds of PCBs are possible. However, digital printing is currently explored, but there are still many challenges.

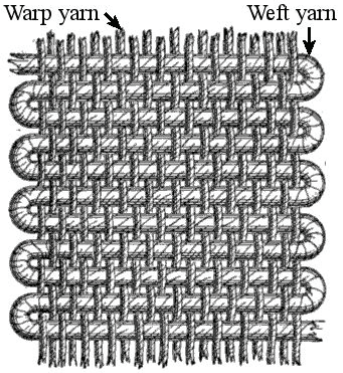
Coating. With conductive coating (again, either on metal or polymer basis), all materials can be made conductive while a non-conductive coating can serve as an insulator of conductive surfaces. That is a possibility to process textiles with the standard machines and techniques and add conductive properties later on.

Connection to traditional electronic components. Still, at least some kind of power supply made of traditional material is necessary to work with active smart textiles and often a great number of electronic components are used. The connection between traditional electronics and yarn or fabric is a big challenge in smart textile research. Depending on the textile qualities of the yarns, in some cases soldering can be used to connect both parts (we tested some yarns if they possess that quality—see the results in section 6.3.2). When the textile is heat-sensitive and therefore cannot be soldered, gluing with conductive glue is another way to connect the two different materials. Some projects use and even weave very thin cables instead of yarn, because they are insulated and can be soldered easily (cf. [Martin et al., 2004]).

Conductive glues are adhesives filled with conductive materials, for example silver. Due to the silver, the conductive glues are very expensive. Some glues use carbon instead of silver to reduce the costs. There are also conductive adhesive tapes made from copper, aluminum or nickel.

It is also possible to sew or weave components directly into a fabric (cf. section 6.3.2), but the connection is not stable and durable. Other possibilities to connect electronics and textiles include conductive textile connectors (e. g. snap buttons) [Post et al., 2000] that can be riveted to conductive yarns and fabrics or "crafty" solutions [Buechley, 2006] that bend electronic components to sew into the originated holes and eyes. The connections require a great amount of manual labor as up to now there are no textile machines on the market to help with the process.

Figure 5.14: Warp and weft yarns in weaving, adapted from The History and Principles of Weaving by Hand and by Power,1878, S. Low, Marston, Searle and Rivington, London. Released into Public Domain.



5.2.4 Applications and Projects

We just introduced a range of new materials and of using traditional textile techniques to integrate these new materials into textiles and clothing. It is easy to see that the new materials can enhance traditional clothing in its ability to listen, think and speak—to make them more interactive. In our concept, we pointed out the importance of familiarity. While textile materials, especially smart materials, might not be familiar to most people,

clothes definitely are. Therefore, we will take a look on how we use and create meaning with smart textiles and we will focus on the applications in respect to art and interaction design. These examples are different from wearable computing in their placement in social practices and the context of fashion as well as in their interaction and aesthetic qualities.

We mentioned before that at the moment three market segments for smart textiles exist: sports and medical wear, work-wear and accessories. We apply a slightly different perspective to categorize smart textiles here—namely the user's perspective. What functions, social as well as technical functions, does an application have? By analyzing current examples of such applications, we could identify the following categories or topics that are not strict but often overlapping:

1. The Body as Display or New Output Devices
2. Functionality, Sports and Medical Applications
3. Awareness Over Distance
4. Mobile Communication
5. Mood
6. Silhouette
7. Games, Play, Fun
8. Criticism
9. History and Morphing
10. Control.

The similarity of the projects lies not in the degree of interactivity. All projects focus on aesthetics instead of on function alone. There is also a similarity in the way the examples take up social practices evolved around clothing in our culture long before smart textiles existed. Then the designers build the applications around these practices. They start from the practices embedded in our everyday live, in clothing and fashion and transform them. Therefore, we argue that the examples chosen implement a design considering (social) context and even more—implementing the approach Dourish proposed in his embodied interaction framework.

The Body as Display or New Output Devices. Textiles are becoming smart and can also act as new output devices and display information of any kind. Using the body as display to irritate or to simply make statements is not a new idea in fashion and art. The boundaries between body and display can blur, smart textiles are not the only opportunity to point that out. Still they offer a lot of new possibilities. They can change their display dynamically, according to the wearer's context. Moswitzer and Jahrman, for example, created the Pong Dress[26], a shirt with a 4 x 6 resolution matrix and game

[26]http://www.ludic.priv.at/

controllers. People can play Pong by staring at the wearer's body all the time. Debatty from we-make-money-not-art[27] suggested that the inspiration of the Pong dress came from Export's Tap and Touch Cinema[28], where the artist taped a box to her naked chest and invited people to join the cinema. This installation was made in 1968.

Buechley and Eisenberg [Buechley and Eisenberg, 2007] made what they called *LED clothing*, shirts and bracelets containing several LEDs (up to 150). These items can be programmed with a mobile device to display various patterns, e. g. the *game of life*. Tachi [Smith and Trophan, 2005] highlighted the topic differently. He tried to make the wearer's body invisible by creating a camouflage dress, called the Invisibility Cloak that used a light reflecting material in the cloak as projection space for photo shots of the environment behind the wearer. Wearer and the environment merged for the eyes of the viewer.

On the commercial market, we can find a number of t-shirts, base-caps and bags that display information with electroluminescence foils. An example is the Wi-Fi detector shirt that indicates open wireless networks in the wearer's surrounding[29].

Functionality, Sports and Medical Applications. This is an area that has already produced market-ready products, opposed to most examples in this chapter that are prototypes. Most smart textile research takes place in this area. A lot of large companies are investing into the field. The first smart textiles application for consumers was a skiing jacket from Phillips and Levis [Pahl, 2004] that integrated a MP3 player (the product already disappeared from market again). Winter and snowboard and skiing wear are often used to include electronics, for example in the commercially available Phillips and Levis jacket. We think, the reasons are that these garments offer a lot of space to "hide" electronics and are very expensive anyway so people might spend more money to have additional functionality.

There are numbers of projects connecting MP3 player controls to garments. While the control is textile, the connection to the player often stays cable based. More advanced applications include Nike and Apple's iPod sports kit[30] in which an iPod receives and processes data from a sensor in the shoe. The wearer uses the kit while running and gets real-time data of, e. g. her pace, or calorie consumption.

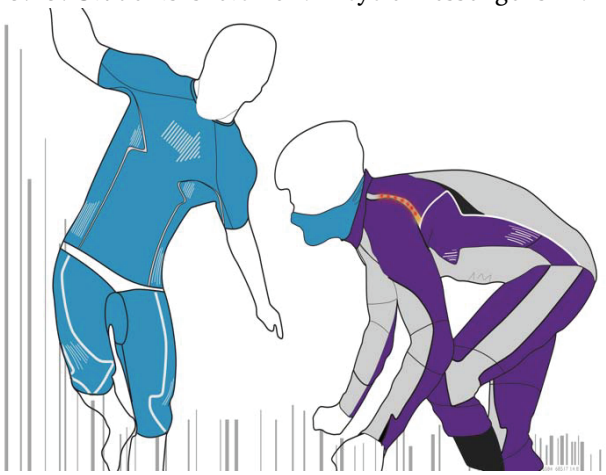
Another area that is popular at the moment, is bags that gain power through flexible solar panels and can be used, for example, to recharge the wearer's mobile phone while she is traveling.

In medical applications, body monitoring is an important topic. Products, e. g. the LifeShirt[31], are designed to monitor the wearer's body constantly, to allow more precise diagnoses. The textile material makes them more comfortable compared to traditional medical instruments. For example, instead of wearing a plastic sensor on a belt strapped around the chest, pulse monitors can be integrated into underwear people wear anyway

[27] http://www.we-make-money-not-art.com is a well-known blog on media arts
[28] http://thegalleriesatmoore.org/publications/valie/valietour3.shtml
[29] http://www.thinkgeek.com/tshirts/generic/991e/
[30] http://www.apple.com/de/ipod/nike/
[31] http://www.vivometrics.com/

(cf. Numetrex's heart rate monitor products[32]). The wearer cannot feel the difference between the pulse monitor and the textile. A different approach, but also an example of sports related smart textile ideas is the interactive brake light for bicycle messengers that reacts to pressure sensors on the wearer's hand. The project was developed by students during an EduWear workshop in Sweden (see the design sketch in figure 5.15 as an illustration).

Figure 5.15: Students' sketch of a bicycle messenger's brake light.



Awareness Over Distance. Smart textiles are used to communicate over distance or to be aware of what a distant person is doing, for example, whether she thinks about the partner. The concept of *awareness* refers to the perception of something in the environment. It can refer to the computer being aware of the user's activities or surrounding, but it can also refer to the user's awareness of other users and their activities.

Cute Circuit created the Hug Shirt[33] which is a great example of this kind of applications. The shirt has embedded sensors and actuators. It senses when one shirt hugs (the wearer performs a hug-like movement), and sends the signal via Bluetooth and a Java enabled mobile phone that has the HugMe software installed to a partner shirt. The partner shirt will simulate a hug with its actuators (e. g. small vibration motors). What we call devices for awareness, Baurley referred to as "tools for remote interpersonal communication" [Baurley, 2004, 277].

Our Smart Dance clothes can also be used as awareness devices. The clothes are two partnering shirts and trousers. The shirts have accelerometers attached to the arm and sense the movement of the wearer through these sensors. The movement is displayed on the shirts, but can also be recorded and sent to friends who can then receive the message as light pattern on her mobile phone or as a pattern for her own shirt. As display, the gent shirt uses three LEDs hidden under the shirt's collar to emit a rather ambient light. For the ladies shirt, the LEDs are placed under a layer of thin translucent fabric and glass fibers prolong the LEDs to be visible as light points on the wearer's arm (cf. figures

[32] http://www.numetrex.com/
[33] http://www.cutecircuit.com/now/projects/wearables/fr-hugs/

Figure 5.16: The Smart Dance clothes react to movement, e. g. dancing or jumping.

5.16, 5.17, 5.18). The trousers that belong to the Smart Dance shirts, emit light through electroluminescent wire sewed into channels of translucent fabric.

The Smart Dance shirts take up the practice of sending small messages (that we saw in the examples on how children and young people use mobile communication devices in section 5.1.1) and allow interacting through gestures. A gesture for saying "Hi" can be a wave, for example, just as in the real-world. We take up an everyday practice for greeting and connection between people. (Young) people can display the greetings they received on the shirt; they are displayed as light patterns. The user can as well see the greetings on her mobile phone, if she does not have her own Smart Dance shirt.

Technically, both shirts have accelerometers and LEDs that connect to an IC based on Arduino technology (see section 6.3.2 for details). The Arduino board reads data from the accelerometers and sends it to a computer (that could also be a mobile phone) via Bluetooth. On the computer, a Processing (again there are more details in section 6.3.2) program is waiting for data from Arduino and translates the incoming data into a simple visualization. We decide whether a wearer moves the arms upwards, downwards or horizontally. The visualization is recorded as an animated gif that can be shown on a mobile phone or as a mood message on an instant messenger or in a profile of a social networking site (cf. figure 5.19).

Mobile Communication. Fashion is always about communication. With communication related to smart textiles, we refer to mobile communication or security functionality

Figure 5.17: The Smart Dance shirts react to the movement with changing light patterns.

integrated into textiles. An example of a kind of non-communication is the No Contact Jacket[34]. The jacket has an electroshock device integrated that the wearer can activate if needed. A less drastic project that used smart textiles for communication was, for example, a glove by a Swedish research and development team that integrated microphone and speaker to facilitate mobile communication in cold temperatures [Berglin, 2004]. Berglin's speaker glove was an example of work wear. Other example applications include radio devices integrated into work wear, e. g. for rescue workers, who need radio contact in extreme environments.

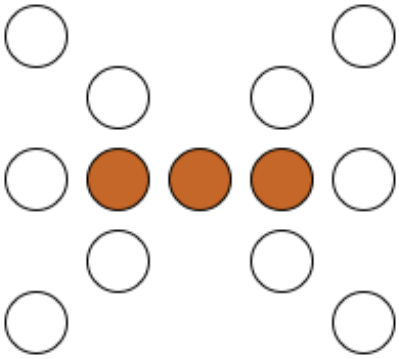
Mood. Clothes and accessories can show the current mood of the wearer. Smart textiles allow an automated detection of a mood by using sensor inputs. This automated detection is not necessarily desired, a user-controlled mood display is an alternative.

Silhouette. The body silhouette is an important topic in fashion. The new school of smart textiles uses the silhouette to discuss the topic personal space. For example, Berzowska[35] and Almeida (see [Seymour, 2008, 153]) use balloons hidden in the garment that can be inflated once the user presses a certain button.

[34] http://www.no-contact.com/
[35] http://www.xslabs.net/work-pages/inflatables.html

Figure 5.18: The wearer can record and send a visualization of the light patterns to a friend's mobile.

Figure 5.19: The resulting visualization of the wearer's movement on the horizontal level. When the wearer throws her hands in the air the light points move upwards; when the arms are in a downward position they will move down.



Games, Play, Fun. The already mentioned Pong Dress is a game that is played on a body display. Other forms of gaming are musical instruments that appear in textiles. For example, the Musical Jacket [Post et al., 2000] where the user could hit a textile keyboard and a micro-controller processed the data as midi-signal and played a tone on speakers integrated in the jacket. Glaister and Shakallis [Smith and Trophan, 2005] had

a similar idea. Their Barcode Textiles can be read with a modified mobile phone and translated into ring tones. The Tic-Tac-Textiles [Redstroem et al., 2005] is a tablecloth based on thermochromic ink. With cups filled with a hot beverage, e. g. coffee, the user can play a Tic Tac Toe game. The tablecloth is meant to be placed in areas where people have to wait.

Criticism. Smart textiles also bear risks, not only surveillance and intrusion into private space might become easier through these new technologies, we also expose ourselves to electromagnetic pollution and consume a lot of energy through the new devices. Fashion Victims[36] [Lee et al., 2005] produced a bag that itself is a smart textile, but on the other hand made the invisible electromagnetic radiation visible. Their bag "bled" (emitted red-colored ink) and the intensity of the bleeding depended on how many phone calls were made in its environment—technically how much electromagnetic radiation is measured.

History and Morphing. In *haute couture* we find few examples of smart textiles. Chalayan is one of the few well-known fashion designers who work with these new technologies. In October 2006 he presented transformer dresses that changed into different styles from different fashion episodes [Zainzinger, 2007]. The Hour Glass dress caught most media attention (probably due to the fact that the model's whole dress morphed into a hat leaving the model naked). The dress Chalayan made with the technology company 2D3D[37] had a number of motors attached that closed zippers and changed hemlines. Another well-known fashion designer that works with smart textiles is Waldemeyer[38], his creations rather fit into the body as display area.

Berzowska launched a project called Scorpions[39]. In the Scorpions project, they create garments that are out of control of the wearer, changing and morphing due to computer programs that control shape memory alloys (see figure 5.20, for the dress called GLUTUS). Another idea concerning history is a dress that remembers the wearer's past actions. Our Smart Dance shirts are also able to collect the movement of the wearer during a day and then translate the moves into light patterns displayed on LEDs in the collar.

Control. Scorpions—that we mentioned just before as an example of morphing—is also a project that broaches the issue of *control* in clothing. Most of the time we are in control of what we wear (at least as adults). We choose our clothes and they hopefully behave in a way we can foresee. Smart textiles can be different in that respect, because through the computational power they can have their own artificial intelligence and behave according to algorithms. The wearer is not longer in control, she is rather worn by the garment. In the Scorpions dress GLUTUS (cf. figure 5.20) a leaf-like "hat" is starting to consume the wearer.

[36] http://www.fashionvictims.org/
[37] http://www.2d3d.co.uk/
[38] http://www.waldemeyer.com/
[39] http://www.xslabs.net/skorpions/

Figure 5.20: Scorpions, an XS Labs project by Berzowska and Mainstone. The dress called GLUTUS has a leaf-like hat (that one can see in the left picture) that moves with shape memory alloys and starts to "consume" the wearer. Photograph by Stinghe. Licensed under Creative Commons Attribution Non-commercial License

5.2.5 Context and Creation and Sharing of Meaning

The examples used here can all be seen as *interactive textiles*, when we use our definition of interaction. On the other hand, that is also true for applications from wearable computing. The level of interaction of our examples is very different. Some of the projects are more like traditional clothes, in the sense that they rather speak than think or listen. Examples of projects with an emphasis on speaking include Buechley's LED clothing, Chalayan's Hour Glass dress or the Inflatables. The new way of interaction with these examples is to use them as novel output devices. In these projects there is rarely an input, they do not listen. A combination of input and output, of listening and speaking are our Smart Dance Clothes that listen to the wearer's movement. Other projects do a great deal of thinking, for example the Barcode Textiles, that translate a printed pattern into a melody, or the Scorpions, where the garments have their own artificial "character".

We now take a closer look at practices around traditional clothes, and how these smart textiles projects relate to them. When looking at the relations between using traditional clothing and the smart textile examples for a certain function, we see a lot of similarities. Using the body as a display is common practice not only—but especially—in sub culture and youth culture. T-Shirts with (for example political) statements use the

body as display or for criticism in pop culture. Also, wearing brands can be considered as a practice of this kind.

Awareness over distance in fashion was longtime expressed by friendship bracelets worn to show the world who one was friends with, how many close friends one had and as a reminder to think of your friends while they were away. Of course, these pieces of clothing stood in relation with certain practices and actions—the act of giving a bracelet to a friend for example.

To display one's mood in clothing has very strong tradition, too. To wear black while being in mourning is an example, as well as various kinds of party hats or clown's noses to show a festive mood.

To discuss meaning making and bringing in one's context in smart textiles, we have to take a closer look at meaning creation and the role of context in traditional clothing that we just started to mention. The process of meaning making is a complex one, as the meaning of course relates to the context the wearer is wearing the clothes in and the combination of clothes and accessories.

First of all, we asked twenty-nine children and young people that were participants of our workshops (cf. section 6.1.1), how important clothes and fashion were to them and what role they played in their life. Figure 5.23 shows details of age and gender of the children and young people we asked with a questionnaire in the beginning and at the end of the workshop. Most of the participants were male.

Our question was: What does clothing mean to you? The children and young people could score different answers with four values between "I totally agree" to "No, not at all". Figure 5.21 and figure 5.22 show the answers the participants gave us in more detail.

What we can learn from the data is that the children and young people we asked saw clothing as something important to them. The function of clothing is rather to act as a second skin, protection and self-expression than as a means to show belonging to, setting apart from groups or to show off wealth. That is at least, how the children and young people we asked, answered it. We can show with this data that fashion is important to the youngsters and that they are confident in admitting it. We can also see an overlapping with the topics artists and designers worked on in the example projects we mentioned before.

Research on meaning making with clothes, on the other hand, places emphasis on the role of belonging to and setting apart from different groups. We mentioned the concept of bricolage before, when defining the design process as assembling different meanings to a new object (cf. chapter 3). In cultural studies, sometimes a semiotic view on clothing and style is applied. The objects that are worn and that build a bricolage in their combination are signs that can be contextualized and re-contextualized [Hebdige, 1979] in a dynamic process rather than in a fixed set of signs. According to this theory of *style*, everyday objects can be taken and made to signs that have a meaning or they can change their meaning due to the process.

We mentioned wearing black as a possibility to show one's mood namely being in mourning. That is only one meaning wearing black can have. We can also wear black

Figure 5.21: What does clothing mean to you? The participants perceived clothing rather as a second skin and as protection than as an underlining for their personality.

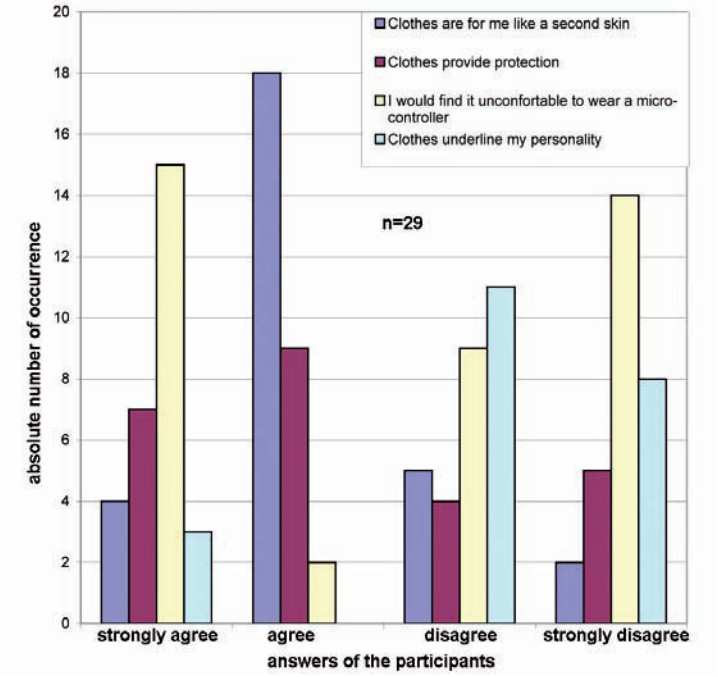
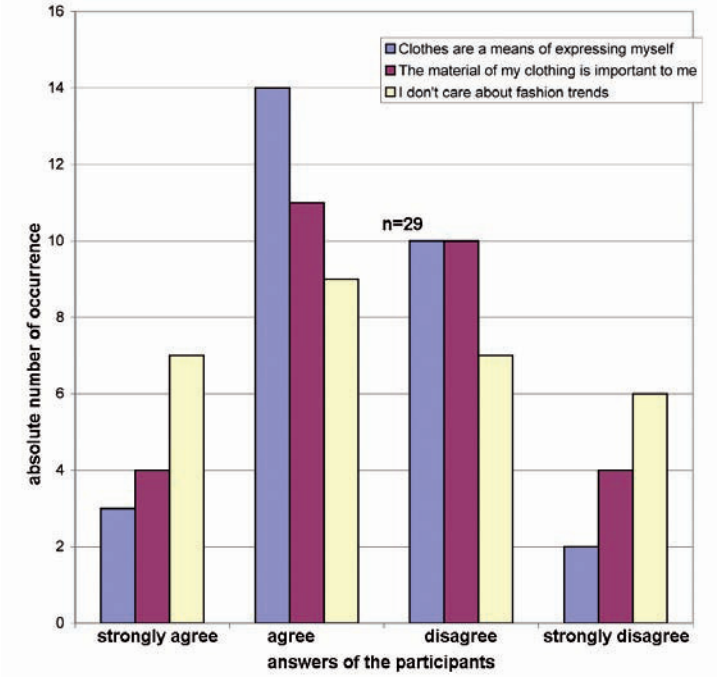


Figure 5.22: What does clothing mean to you? Self-expression through clothing played a role for many participants on the one hand, on the other hand a lot of the participants also disagreed.



to show our affiliation to a certain group or subculture, e. g. artists, or *gothics*. To wear black and use items related to mourning and death—, e. g. pale make-up or religious

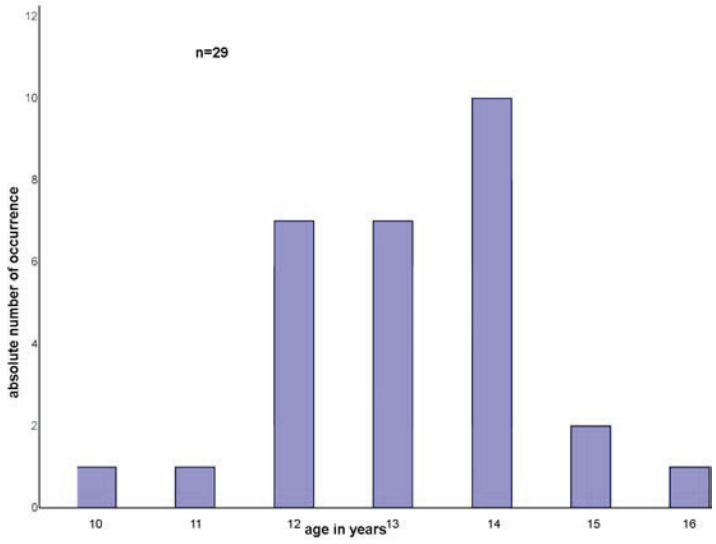
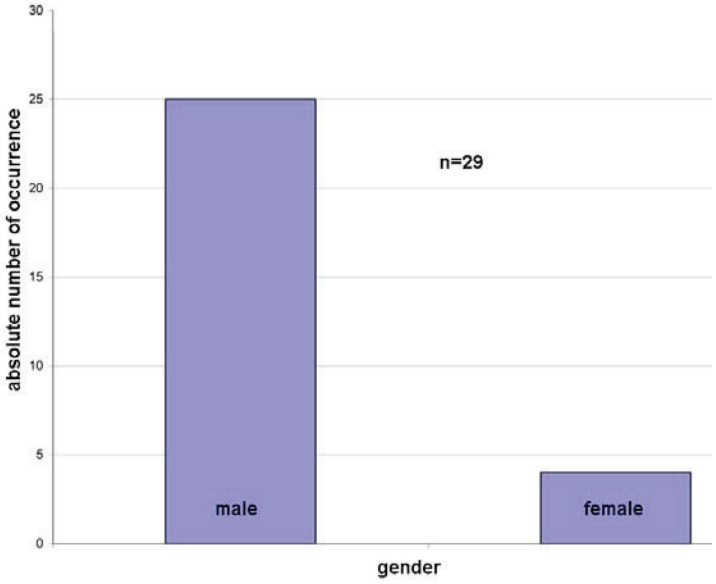


Figure 5.23: Age of the children and young people we asked what clothing meant to them.

Figure 5.24: Gender of the children and young people we asked what clothing meant to them. Most participants were male.



symbols—in the gothic subculture, is a way to turn them into symbols of a certain lifestyle. The meaning of clothes and accessories arises on a different level, because the common practice in main culture (here: to wear black in mourning) is transformed by the subculture.

The body silhouette which we also discussed in the examples is quite similar. Silhouette was important all the way through fashion history and subcultures transform this meaning. According to Edthofer, the body silhouette for example in Hip Hop culture is used to make one look more dangerous on the streets [Edthofer, 2003]. That is her

explanation why the *baggy style* became so famous. If one looks at the Inflatables from Berzowska or Almeida, one can see the smart textiles designers taking up a different concept of silhouette closer to the subcultural meaning in Hip Hop: the one of personal space. The idea is to enlarge one's body silhouette to create more personal space and communicate the need for boundaries.

Clothes traditionally have and show a history of wearing. That may simply be abrasion of the material or personal memories connected to the garment, but can also be a fashion statement. Some garments, e. g. fan merchandise, for example rock bands' tour shirts, or sport competition shirts, make the statement that the wearer was there and took part in a certain event.

We can see by relating the smart textiles examples to traditional clothes that meaning creation through "normal" clothes already is a complex process and the meaning depends on the cultural context in which the bricolage of clothes, garments and accessories is worn—the style depends on the context. This transformation process can also be seen as appropriation, especially young people create new and unintended uses and meanings of clothing and accessories (a great example of such an appropriation is the early seventies' *punk* style where young people started to wear safety pins as accessories).

When we look at this bricolage perspective on meaning making and clothes, we often find a notion of the term that refers to a set of rules and codes. These rules are context-dependent or—as we mentioned before in Hebdige's approach—are constantly contextualized and re-contextualized. Referring to Schütz, we could also say that clothes will have different meanings, depending on the province of meaning. What is an appropriate dress in the world of leisure might be interpreted differently in the work world.

> As such, fashion behaviour varies with context. Accordingly, fashion has no absolute or essential meaning, rather the clothes-body complex operates in ways appropriate to a particular habitus or milieu. [Craik, 1994, 10]

From the phenomenologists' perspective one could argue that the meaning is not only dependent on a certain context, but again that there is no such thing as a fixed meaning that can be attached to a piece of cloth worn in a certain situation. We can rather see fashion, clothes and accessories as a zeug. The meaning then arises dependent on and from our actions and our intentions. We pointed out that children and young people perceived clothes as something they use to express themselves. Choosing what to wear is influenced by the actions in the world, if the wearer's intention is standing out of the crowd and expressing oneself, showy clothes might be fine and feel good while when the wearer aims at hiding and staying unnoticed the showy clothes will suddenly become troublesome.

Smart textiles, or rather examples of applications in the field of (interaction) design and smart textiles, can relate the novel materials to the context of fashion and familiarities to design around certain practices. Still, from a phenomenologists' perspective the textile artifacts will always be interpreted in connection with actions in the world and their meaning depends on the action. How the new possibilities of smart textiles will change actions and intentions on what to use clothing for, is not yet known.

As a conclusion, we showed that practices around clothing are familiar for children and young people and that smart textiles can draw on these familiarities—the example projects showed possible ways on how to do it. (Smart) textiles are also a flexible medium—at least in the way children and young people use it: Through combination of signs children and young people can make their own meaningful project. Smart textiles feel and look nearly like traditional textile material, children and young people can combine the smart materials with their own clothes that have a meaning for them and create new bricolages through the process. We saw that often sub communities define themselves through their clothing (although the children and young people perceived it differently). A community centered on the design of smart textiles could and would possibly create their own style.

5.3 Participatory Software Design with Children

We concluded in our concept that it is necessary to relate to the users' practices in their real context, instead of the user's behavior in an artificial scenario, e. g. in a research lab. The question on how to get information about the users' social practices remains. Dourish referred to methods, where designers go into the workplace of the people they design for, to get an impression of the context the users act in. As our aim is to design constructionist learning environments, we would have to go to scenarios where children and young people learn, to deploy such methods. Up to now, we used studies and research (for example the JIM study and theory on clothing) as well as studying the users' usage patterns (by analyzing YouTube) or simply by asking users to find out how they use tagging and (smart) textiles in their real life. We consider these methods as suitable to get an impression how and what technologies children and young people use. A further argument for deploying such methods is that the users' attitudes and actions change during the development process.

In interaction design for children, we find a whole research field dealing with the question, on how to involve children and young people into the design process. Especially with very young children methods commonly used with adults have to be adjusted. Children are harder to question about their behavior. Also, they do not always have a fixed setting the designer can go into to observe, as for example in the adult's world, one can go into a workplace scenario. To use a school scenario might be counterproductive, because often there is a lack of playful and engaged learning at school (cf. [Schelhowe, 2007, 108]). Children also cannot articulate their ideas and preferences precisely in the adults' way. Another problem of designing with children and young people is the diversity. Age, gender, stage of development and social background, for example, play a great role when designing for this target group. In our context-dependent view that is not surprising.

In this part we briefly introduce methods from interaction design for children to include them into the design process. We often find these methods referred to as *participatory design* (PD) methods.

Druin is a pioneer in participatory design with children and young people. She has worked with children and young people as active and equal design partners from 1999 on and developed a method called *cooperative inquiry* [Druin, 1999] which combines the Scandinavian participatory design approach with *contextual inquiry*. Contextual inquiry is an interview method to gather data typically from users while they are performing their tasks in their contexts. After the contextual inquiry session, the children and young people design low-level prototypes with materials like crayon, Play-Do or cardboard, because in Druin's opinion it is easier for children and young people to express their ideas with this kind of material, instead of communicating them verbally.

Cooperative inquiry according to Druin [Druin, 1999] is divided into three phases:

1. *Contextual inquiry*: Children observe other children and young people and also adults by using existing products. They note their impressions about where the users have problems. The novelty of Druin's approach is that children and young

people take up the position that is usually reserved for researchers. As an outcome of this first phase of the design process, the designer gains ideas how children and young people perceive interactions with the artifacts.

2. *Participatory design*: Children and young people develop and express their ideas for a new (software) product with tinkering materials and storyboards. There are different *design activities* to support the different types of children that can be used in the second phase of the design process.

3. *Technology immersion*: Children and young people can interact with new innovative technology that is usually not available to them. For example, smart textiles are something that is new and fascinating so we should present it to children and young people in phase three.

Cooperative inquiry is the most popular approach for participatory design with children, according to Williamson [Williamson, 2003]. Originally it was used to work with children aged between seven and eleven. Modifications of the method, e. g. the *mixing ideas* method [Guha et al., 2004], exist to make the method also usable with younger children. Although the method is popular, there is criticism by different researchers. Cooperative inquiry is considered as very time-consuming and therefore hardly adaptable to a lot of scenarios. For example, Jones et al. [Jones et al., 2003] reported problems with using the method when designing screen-based systems, because storyboards were unknown to the children and young people they worked with, and it was often hard for the participants to understand the translation from a storyboard to screen. We also like to mention that a lot of examples of cooperative inquiry take place in research labs, where children and young people come over for a *design session*. The context in which the researchers observe children and young people interacting with technology is therefore very different from the real-world setting in which the young users would use a final artifact.

Another popular approach for participatory design with children is *children as informant designers* (cf. [Williamson, 2003], [Scaife et al., 1997]). This method sees children and young people not as equal partners, but as informants. They are questioned about their experiences with existing technology and in later design phases with the prototypes. It is not a linear development process. The children and young people develop low level prototypes (made of paper, cardboard, play-do and similar materials) in parallel with the adults' development of high-fidelity prototypes that are constantly tested and redesigned (cf. [Scaife et al., 1997]).

Both approaches, cooperative inquiry as well as children as informant designers, first gather impressions on the social practices children and young people have developed by contextual inquiry or other forms of interviews and then give room to the children and young people to express their ideas on new and better designs for technologies.

5.4 Summary and Conclusion

Coming from studies on how children and young people use the Internet and related digital media, we discussed applications and concepts summarized under the umbrella term social software. We showed that social software and especially tagging are technologies and practices familiar to children and young people, some usage patterns are also different from the adults' ones. Children and young people use tagging systems like YouTube as social networking sites to connect to each other. Tagging as well as other approaches in social software, e. g. social navigation, offer possibilities to integrate one's context and to design open places.

When we discussed research on the question of meaning making and sharing meaning in tagging systems we had problems applying a phenomenologists' notion of meaning and found a representational perspective predominant. What we can say is that tagging can be used by communities to create a common vocabulary, though. Communities and sub communities—learning communities as well as others—can design their place by developing their own vocabulary and their own practices how to use it.

Clothing and fashion are another domain very familiar to children and young people and they (especially young people in youth and subculture) take up traditional clothing to create a lot of ways to express themselves and integrate their context. Appropriations—finding new and unintended uses of clothing and accessories—play an important role for meaning creation among the wearers; we can therefore say that clothing is a flexible medium. Smart textiles can serve as a digital medium that relates to and supports these practices as we showed by discussing examples from the domain of the arts and interaction design. We ended with pointing out that the meaning of clothing and fashion rather arises out of intentions and context, instead of being fixed and "attached" to a piece of clothes.

As a method to involve users (especially young users) in the design process from the beginning on, we introduced participatory design methods that were adapted for designing for and with children and young people. We consider cooperative inquiry as an approach to work with in the construction part of this thesis, because it is a method that can help to implement our design principles one (start designs from familiarities) and two (develop in a participatory design process).

As a conclusion, we consider the application domains tagging and smart textiles as suitable to be taken up for the implementation of an environment that puts constructionism into context. In the next chapter, we will discuss how we take up the technologies to create a learning environment as a showcase for our concept.

Chapter 6

Designing the Learning Environment: EduWear and Amici

With the EduWear construction kit and Amici as programming environment, we aim at showing how to take up tagging and smart textiles for a constructionist learning environment to implement our concept.

The development of the construction kit partially took place within the European project EduWear[1] that has been funded with support from the European Commission. The European project took place from 2006 till 2008, coordinated by Prof. Dr. Heidi Schelhowe, head of the research group DiMeB at the University of Bremen, in cooperation with the Comenius University, Bratislava[2], Slovakia; the St. Patrick's College, Dublin[3], Ireland; the University College of Boras[4], Sweden; the University of Dundee[5], United Kingdom and Extended International Consulting Services. Ltd., Hungary.

Especially the textile parts of the EduWear construction kit were developed in joint work with our Swedish partners and have been manufactured in Sweden. Many ideas and techniques where developed together; e. g. color codes and circuits of patches by the author and the actual textile production by our Swedish partners. The author's main contribution is the programming environment Amici and the hardware development as well as studying and organizing workshops, design sessions, tests and interviews.

We will start discussing EduWear and Amici by introducing the first sessions of the design process that we used to gather first ideas and requirements in collaboration with our young users. Afterwards, we introduce the EduWear kit in its different iterations.

[1] http://www.dimeb.de/eduwear
[2] http://www.fns.uniba.sk/en/about/
[3] http://www.spd.dcu.ie/
[4] http://www.hb.se/ths/
[5] http://www.dundee.ac.uk/

6.1 The Design Process

We worked in a participatory design process inspired by Druin's cooperative inquiry with children and young people. Most of the children and young people we recruited from robotics and similar workshops. We will introduce concept and structure of these workshops in more detail in the next section, because we refer to them as our design and also evaluation setting. The workshops are our real-world context in which we observe children and young people.

6.1.1 Design Workshops Structure and Approach

The research group DiMeB regularly organizes workshops for children and young people aged between nine and fourteen. The range is quite wide and diverse. Within the workshops, children and young people work in small subgroups to construct and program smart physical artifacts. These artifacts can have every possible form depending on the character of the workshop: from robots to smart clothing. The workshops are sometimes announced with a certain topic, e. g. "Body and Movement", or "Interactive Fantasy", and take place in different settings: within the school environment, in youth hostels, public libraries or stretched over some days in the research lab. When the participants visit the research lab five days in a row, it becomes something normal for them. The children and young people mostly work by themselves on their own ideas in their subgroup, but tutors support them. Some activities take place in the large group with all participating children and young people. The workshops, concepts and experiences are discussed in greater detail by Reichel et al. [Reichel et al., 2008] or Dittert et al. [Dittert et al., 2008]. The workshops differ from another, but there are some constraints to all of them. We discuss two different workshop implementations in more detail in chapter 7 on evaluation. During a workshop, the children and young people will typically go through different phases or *stages*:

1. *Fantasies about technology:* In this phase, children and young people brainstorm and associate ideas they have related to technology and the specific workshop's topic (if any). We use storyboards, sketches and cards as design activities to support children and young people during the process.

2. *Introduction to hard- and software, and material:* Children and young people can see and touch the material and learn what is possible to do with it. These materials can, for example, be smart textiles, the programming environments, robotics, electronics and various "traditional" textile and tinkering materials. The introduction to the material can take place in subgroups, but also in the whole group through presenting materials and examples.

3. *Development of project ideas through imagination:* Usually the children and young people start with a sketch of a first rough idea and develop their project iteratively from there on. Sketches are used for the design, but many children and young people start with building something right away and then come back to a planning

phase later on. The ideas of each child have to be discussed in the subgroup, so that the subgroup negotiates one common project.

4. *Conception, construction and programming:* The cycle of conception, construction and programming is the core of all workshops. In this phase, the children and young people actually make and program their artifact as a working high-level prototype. They build something and write a program, try out the program and then debug or change construction and code. Or they change as well the "story"—the idea they work on.

5. *Public presentation:* At the end of each workshop, children and young people present their work to a public—mainly to their friends, teachers and parents. Presentations can be in different forms, they could be a fair but as well a show on stage.

Next to these workshops, DiMeB supervises a group of children and young people that already took part in several workshops and was especially interested. Now, they come to the research lab weekly to work on more advanced and ongoing projects. We refer to this group of children and young people as the *expert group*. With the children and young people of the expert group, we also discuss, develop and test new technologies. The idea of the expert group is related to the computer clubhouse approach that we introduced in the section on examples of constructionist learning environments (cf. section 3.8.4).

The technology used in the workshop depends on the topic that is announced and on the (age) group of children and young people. Lego Mindstorms, Handy Crickets and RoboLogo —Grüter's adaptation of the open source programming software Jackal[6] [Grueter, 2006]—for Handy Crickets as well as the EduWear construction kit are used.

6.1.2 Contextual Inquiry

We started the design process for the EduWear construction kit and Amici from the study we did about the way children and young people use tagging systems with the example of YouTube in section 5.1.5 and contextual inquiry sessions. While we introduced the method contextual inquiry in chapter 5, we now show how we applied the method in our own work.

We used existing robotics software and hardware (with focus on the software) namely RIS and RoboLogo for two contextual inquiry sessions with children and young people (eight children and young people took part in the roles of test persons and note-takers), using the methods Druin developed [Druin, 1999]. One child and one adult acted as *note-takers* and an *interactor* supported the test person to talk about what she was doing at the moment when using the software. The note-takers were instructed to observe problems with the programming environment. Druin argued for such a method opposed to video-taping, because she noticed that many children and young people start to "act" because of the camera. The adults in our design sessions used the Beyer and Holzblatt method (cf. [Druin, 1999]), a method to note actions the test persons performed and

[6] http://www.instituteofthefuture.org/jackal/

quotes of the test persons in a table. After the observation, the data is analyzed and one tries to identify patterns and roles of the user as well as finding design ideas to improve the existing design. Because we use this method for our usability testing as well (see chapter 7), we give an example of the outcome for a better illustration in table 6.1.

Table 6.1: Example outcome and notation of the contextual inquiry session from 13 September 2006, testing first tagging prototypes based on RoboLogo

Time in seconds	Quotes	Activities	Activity Patterns	Roles	Design Ideas
+00	Hey, what is a "trap"?	Clicks on the tag "trap" (in the automatic tag search)	Browsing of different projects, looking around	Explorer	
+20		Clicks on an element returned from the search	Browsing of different projects, looking around	Explorer	
+21		Clicks on "show"	Browsing of different projects, looking around	Explorer	
+40	What is there on the photo?	Moves closer to the screen	Browsing of different projects, looking around	Explorer	Increase picture size

We found it rather difficult to collect drawings or notes from the note-taking child, so we interviewed the note-taker briefly after the contextual inquiry sessions.

Analyzing the adult's notes of the two sessions, we found out that a problem of the existing software is that children and young people do not know what to use commands for and which commands to choose. Quotes like "Should I use if or ifelse here?" were quite common. As activities, we noted that children and young people either clicked around and moved the mouse over all elements in the software or stopped using the software trying to find out what to do from the peers. This *activity pattern* could be called "searching for the next step". The *role* of the child would be something like "helpless user". Children and young people also had problems with understanding the connection between a port on the physical object and its representation in the software ("what is it? 1 or A?") and reacted in a similarly. These observations are reflected in our vignette (cf. section 1).

We found the results of the contextual inquiry session supporting our hypothesis and assumption that it is hard for the children and young people to bring the programming into context, in that case the context of the physical artifact.

One of the young note-takers had an interesting conclusion: She suggested that (younger) children are not used to click and write, but they rather use drag and drop. She also asked us to provide more examples in the documentation of the software.

6.1.3 Future Design Session

After the contextual inquiry, we collected ideas from the children and young people what to do with smart textiles and also, how to improve the programming environment. Since we worked with five children and young people aged between ten and twelve, we used storyboards and sketches rather than low-level prototypes for the smart textiles applications. The children and young people were old enough to have experiences with storyboarding and sketching. The participants of the design session were all girls. We introduced the task as "imagine clothes and jewelry could be intelligent, what would you do with them?" to gather ideas for smart textiles applications. To find improvements for the programming environment, we said, we wanted to develop a new programming environment that is "better" than the existing ones.

For the design of the user interface, we prepared graphical elements from programming languages for children and young people mixed with typical elements of web design graphics, so that children and young people could extend their own sketches with typical elements. The children and young people created paper mock-ups then. We used a very small range of colors in order to avoid that the children and young people make something that looks colorful instead of concentrating on the function and placement of the elements.

Figure 6.1: Sketches from the future design session. From left to right: a sketch of an improved screen design, the awareness ring and the friend locater.

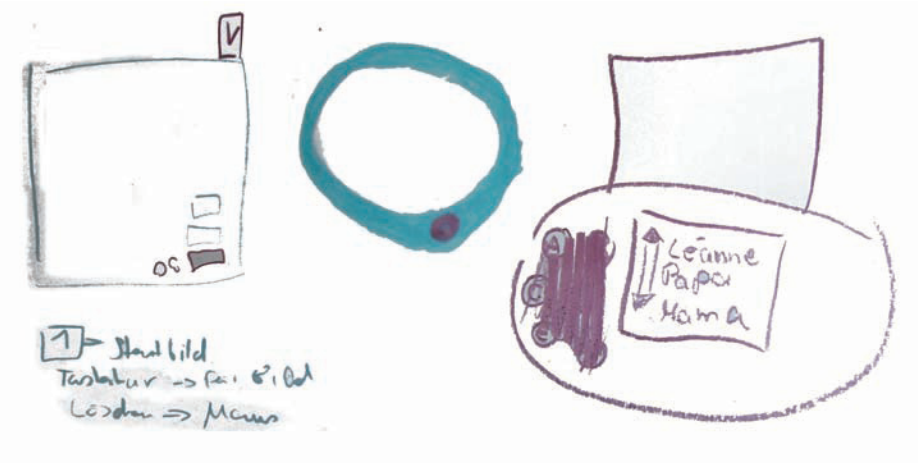


Figure 6.1 shows sketches of the design session: A screen design for a new programming environment, where you drag and drop the elements from the right bottom corner, as imagined by a twelve-year-old girl; an awareness ring that emitted light when "something happened" to close friends and relatives; and the "friend locater", where one could check the current location of friends and relatives as well as their availability for play.

6.1.4 Results

We could see that—in line with our expectations based on the results of several studies e. g. the JIM study we discussed in section 5.1.1—social connections and friendships were

the most important topics the children and young people thought about (see for example the outcomes of the future design session where the participants invented projects like the friend locater). Awareness (as we called it)—to see how the friends are doing at the very moment—is the function the children and young people mostly imagined.

For the programming environment, the children and young people had different ideas. Some mentioned a better representation of the real-world objects e. g. pictures of sensors and actuators and where to connect them. That can again be interpreted as an indicator for the difficulty the young users have in connecting the software representations to their context. The children and young people wanted to drag and drop these pictures around, instead of clicking on an icon and then work with the code generated (as they knew from RoboLogo).

6.1.5 Ongoing Design Process

After the session, we developed prototypes and used them in workshops where we observed how the children and young people interacted with them in participant observations. Then, we eventually changed the design of the prototypes and iterated again. When a further developed prototype existed, we started a more formal evaluation that we describe in more detail in chapter 7.

6.2 Observation: Interaction in the Real-World

For gaining ideas on the interface metaphor of the programming language, we started by observing children and young people in the real-world to complete and change the scenario and find a metaphor. Next to the participatory design process and the study about YouTube, we also did some quantitative observations to get an impression on how children and young people act in the real-world setting. As we want to encourage the users in our learning environment to interact with other people that can even be strangers—the children not necessarily know each other in the beginning of a workshop—, we observed different settings to see where children and young people tend to start interactions.

The setting we design our environment for, is related to the workshops we introduced in section 6.1.1. It is also possible that children and young people keep on working by themselves at home, but in our scenario we assume that children and young people start to get into working in a workshop scenario.

The main question of our short observation was: Do children and young people go to their friends' tables to look at the projects to find help and inspiration or rather to tables with interesting projects?

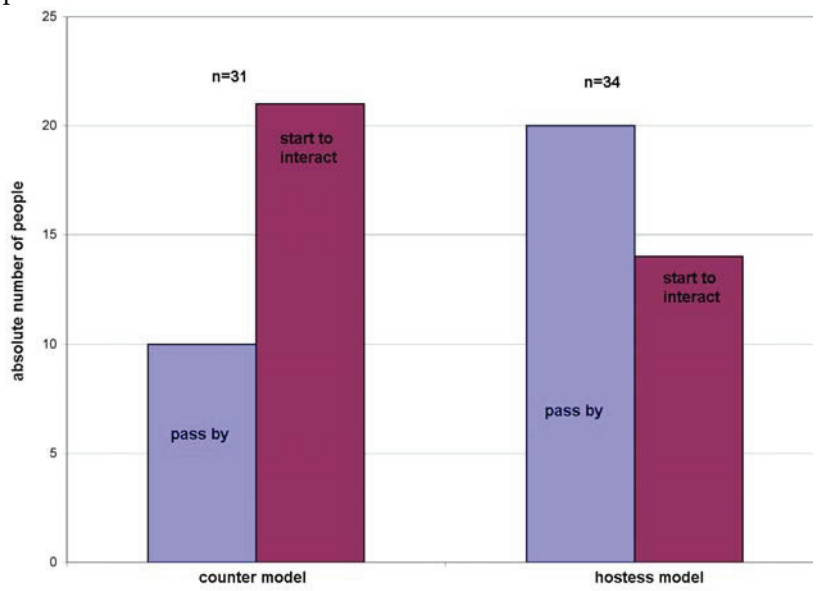
To gain an answer to the question, we observed a fair that we organized at the end of a workshop at school. The workshop took place at the Gesamtschule Lilienthal (a school close to Bremen, Germany) between 9 and 13 July 2007. Twenty children and young people took part; all were aged between ten and thirteen and male.

Children and young people were to present their projects to the public (their friends, classmates, teachers and relatives) in a fair scenario. Using a non-participating quanti-

tative method, we counted how often people started an interaction (defined as visitors starting to talk to the exhibitor) depending on the way the children and young people set up their presentation tables. As different models we identified here: the *hostess* model, where a child would await the visitors in front of the table, smile at them and try to make them interested in her project, compared to the *counter* model where children and young people wait behind a counter where the object is displayed.

The results (cf. figure 6.2) can be interpreted rather as an indicator than as a "proof" but still it was astonishing that in the counter model significantly more people started an interaction compared to the hostess model. In the real-world, interesting objects seem

Figure 6.2: Passing and interacting depending on different models. In the counter model more people started to interact.



to encourage children and young people to start an interaction. To see if this can be transferred into the software metaphor, we implemented and tested different *views* in the software.

6.3 The EduWear Construction Kit

In our study about YouTube, we have seen a connection between the social space or connections between users and their tagging behavior. That is why we heavily rely on tagging to support the social features of the programming environment and the data mining. In the user interface, tagging is not present at first sight, still our models behind rely on tags in different ways as we explain in more detail in section 6.4. Therefore, we call our programming environment *tag-based* programming.

During the participatory design process the construction kit was developed in different versions or went through different iterations. We will give an overview over the basic elements of the EduWear construction kit and then consider the different versions and major changes. The changes did not only improve the development of the construction kit, they also helped us to refine our concept. In this part, we will also introduce the different parts of the kit with respect to their technical development and results that can be generalized.

In general the EduWear construction kit (inspired by constructionist construction kits) consists of:

1. A hardware part that includes a micro-controller, a way to program the micro-controller (an access to the serial or USB port) and a power unit. This part is the *intelligence* of the kit and handles all computation. It allows the smart textile projects to think.

2. *Sensors*. Sensors are the input units of the kit. They enable the artifacts children and young people build to sense certain properties of the environment e. g. light conditions or temperature. The sensors make the smart textiles listen.

3. *Actuators*. Actuators are the output units of the kit. They enable the artifacts to act in the world—to speak. Examples for actuators are beepers or LEDs.

4. *Connectors*. Connectors connect the distances between the different parts. Sensors and actuators have to be connected to the micro-controller, often over long distances. All parts also need to be connected to the garment.

5. *A programming environment*. The programming environment enables the children and young people to make the micro-controller and its sensors and actuators do what they intend.

There are many other materials that make a constructions kit valuable, for example, appropriate tools, tinkering and textile materials, but we mainly focus on the technical development in this thesis. We differentiate between the hardware and textile parts—we call the combination EduWear construction kit—and the programming environment that we call Amici, to make clear what parts we talk about. However, the elements of course belong together and build the kit as a unit. As we pointed out before (in section 3.7), the technology also stands in relation to a pedagogical concept and setting.

We start discussing the development in EduWear with a focus on the conceptual side before we go on and discuss the technical development in more detail. The conceptual development is related to our design principles.

6.3.1 Concept and Iterations

When we refer back to our design principles (that we introduced in chapter 4) the following points were important for designing the EduWear construction kit:

- Start designs from familiarities.
- Develop in a participatory design process.
- Create a shared context and situate the learning process in there.
- Design for wide walls and for flexibility.
- Make something happen in the physical world.
- Carefully adjust between flexibility and familiarity.

We see the principles concerning place and context as more related to the programming environment and discuss them in section 6.4. We already introduced the participatory design process we worked with in EduWear in section 6.1 and discussed the DiMeB workshops as a shared context in which the learning process is situated in section 6.1.1. For the (textile) material, we aimed at starting from familiarities, design for wide walls, make something happen in the physical world, but adjust carefully between flexibility and familiarity.

The (textile) material, the hardware, sensors, actuators and connectors are necessarily part of the physical world, and through the actuators make something happen there.

Textiles as an expressive medium and clothing and fashion are media and domains familiar to children and young people, as we pointed out in section 5.2. By using smart textiles in the EduWear construction kit, we already draw on familiarities. Still, we have to design the parts of the construction kit so that the children and young people can use them to construct their own personally meaningful objects.

In the design sessions, we saw that the children and young people we worked with had friendships and social connections in mind when thinking about new applications with smart textiles, while for the design of the actual construction kit no ideas were developed. To make the different components usable for children and young people, we created what we call *patches*. A patch contains sensor or actuator and the circuit to make it work with the micro-controller. The patches are related to the building blocks metaphor that we already discussed in chapter 3. The building blocks metaphor is familiar for children and young people, for example, through playing with Lego. The children and young people can assemble their project by sewing together the fitting patches.

First versions of the textile patches were rather figurative inspired by (and made mostly by Lena Berglin) Lena Berglin's Spookies (cf. figure 6.3, [Berglin, 2005]). The

metaphor was from the world of toys and world of play. We assumed that this world might be familiar to children and young people.

When testing the figurative patches with children and young people, we observed that most of them rather took single electronic components and made the circuits themselves instead of using the patches. In the interviews after the workshop, the young users reported that the patches "haben nicht in mein Design gepasst" (the patches did not fit into my design) or "die sind eher was für Jüngere" (they are rather for the younger kids). We can see that the users related the patches' design to toys for young children what did not fit into their ideas how the artifact should look. We suspect that although the design was familiar, the associations were not connected to construction material but to ready-made toys for younger children. The design was rather fixed and not flexible; it did not fit into the children and young people's design ideas.

The improved design of the patches lead to a *canvas-style* in the second prototype. The parts were rather neutral (white patches without decoration, see figure 6.4) and could be designed by children and young people themselves, according to their ideas and design concepts. Here, we concentrated on flexibility instead of on familiarity. The canvas-style patches were more often used by children and young people, compared to the figurative patches that the children and young people replaced with electronic parts. The designs with the canvas-style patches ranged from clothes and accessories to bags.

6.3.2 Implementation and Technical Details

As we pointed out in section 3.8 about constructionist learning environments, different hardware systems exist which allow children and young people to build their own smart artifacts. As introduced in the same section (3.8), robotics is a common application widely used in constructionist learning environments although the initial idea was a more flexible tool. The hardware used in construction kits for robotics is hardly appropriate for smart textiles and wearables, because size is not very important, but, for example, driving different kinds of motors is. Therefore, we have concurrent requirements for a construction kit for smart textiles and wearables.

Another important factor for a construction kit is the availability of the materials. For example, the standards of electronic connectors and plugs differ between the United States and Europe. We had a lot of difficulties getting cables for the Handy Crickets in Germany—the American connectors are only available as US-import. Open source hardware can improve the availability very much. When schematics and board layouts of physical platforms are published one can build the hardware oneself as long as the architecture is quite simple and standard components available in different countries are used. One can also substitute connectors that are not available, when building the hardware oneself.

For open source hardware, we have additional requirements, because an active user and developer community is necessary to ensure ongoing development, support and debugging. In a commercial system, the manufacturer would be responsible for that.

To conclude: As requirements for a hardware platform in a construction kit for smart textiles and wearables we identified: good availability, an active community, low price, small size, standard and small power supply and a low weight. The ease of use is a priority; a reduced functionality or a size that is bigger than it could be, are not that relevant. Still, a brick in size of LEGO Mindstorms is too heavy to use. Our first experiences in workshops also showed that children and young people tend to need a lot of actuators, as they often want to control a great number of LEDs independently. The robotics construction kit hardware usually reduced the number of actuator pins.

The cost of a hardware platform is an important factor for a construction kit, too. A lower price improves the availably and we know that smart textiles, e. g. conductive yarns, are very expensive at the current state of research. That is why we tried to reduce the costs of the hardware platform. The costs often depend not only on the micro-controller and the components themselves, but also on the software and hardware needed to program it. Some solutions require an extra tool for burning programs to the chip that is much more expensive than the chips themselves.

Technical Basics: Arduino and Processing

In chapter 3, we discussed learning environments that aimed at implementing constructionist theory. Most learning environments were targeted for children and young people. Similar developments take place in other domains, e. g. creating tools for designers and artists to learn programming.

Maeda [Maeda, 1999] introduced the idea of a combined programming environment and language especially for designers to understand computation in an environment called Design By Numbers (DBN)[7]. The open source community took up Maeda's Design By Numbers and created the Java based language Processing[8]. The languages were meant to create visual output very fast, and the idea is that designers and artists are used to drawing and painting. We can say, they rely on familiarities from the design world. Therefore, metaphors are from the drawing and painting world, e. g. programs are called *sketches* in Processing and you can find them in your *sketchbook*, in DBN programs start with the command "paper" to create a frame to draw on.

Processing was the starting point of a whole family of open source programming languages that share the underlying programming environment. Wiring[9], Arduino[10], and Mobile Processing[11] are the current members of the family which are porting the idea to different output hardware, namely to physical computing (Wiring and Arduino) based on Atmel AVR controllers and to mobile phones (Mobile Processing). The Processing family offers powerful tools, because users can work with the simplified language, but also write code in the underlying standard programming language (Java or C++). We also would like to stress the point that the languages come with an IDE and run without installing, and that the websites of the members of the Processing families are places where the

[7] http://dbn.media.mit.edu/
[8] http://www.processing.org
[9] http://www.wiring.org
[10] http://www.arduino.cc
[11] http://mobile.processing.org/

community of users as well as designers discuss, exhibit projects and help each other. Although coming from another domain, we think that this approach is a great example of designing for "low floor and high ceiling".

We work with metaphors inspired by constructionist learning environments and rely on constructionist theory; still we technically base our work on the Processing family. The main reason is its active users and developers' community as well as the stability of the software (that is still called alpha in the case of Arduino, but is more stable than software designed from scratch would be) and the availability as open source.

We decided (in line with other researchers, e. g. Buechley) to base our development on the open source platform Arduino that is using the Atmel AVR micro-controllers. Arduino includes a hardware platform and software that is based on a number of open source tools. The Atmel chip (IC) Arduino use is either an ATmega8 or an ATmega168. Both chips have the ability to run a bootloader. Arduino's bootloader will run and wait for a signal from the serial port after a reset. That makes it possible to program the IC without using extra hardware.

Different Arduino board layouts exist. The standard board is called Diecimilla in the current version and has an FTDI232R USB interface on board so it can be programmed via USB from a standard notebook without an USB-serial adapter and can be powered from the computer during the prototyping process. Other board layouts include the Arduino Mini, based on SMD components or the LilyPad[12] that allows sewing conductive yarn directly through holes on the PCB (we mentioned it before in section 3.8.3). Arduino Mini and the LilyPad need an extra USB device or a standard Arduino board for programming, and to be powered.

The Arduino architecture (opposed to e. g. Handy Crickets that hide many of the details) makes a difference between power, ground, analogue and digital pins. The digital pins can be used as inputs as well as outputs. The mode has to be declared in the software as we describe in more detail in section 6.4.2.

Textile Materials and Electronics

Textile materials and electronics include connectors, sensors and actuators. Especially the connections between hardware and textiles developed constantly through the process.

Connection between Hardware and Textiles

To connect the elements (sensors, actuators and the IC) it is necessary to lay out circuits on the garment instead of using the somewhat easier way of laying them out on a breadboard. These circuits can be made out of conductive yarn or regular cable. The yarn can be sewn directly to the clothing or prefabricated material can be used for connections. To connect the different parts a basic knowledge of electronics is required. Because most conductive yarns are not isolated, even simple circuits require thinking about the layout carefully, because the yarns should not cross.

[12] http://www.arduino.cc/en/Main/ArduinoBoardLilyPad

As we have discussed in section 5.2, another connection is also required, namely the connections between conductive yarn and electronics that are an important research topic in the field of smart textiles at the moment. For the EduWear construction kit, we experimented with different methods, our objective was to have a connection between yarns and electronics easy enough to be made by the children and young people themselves on the one hand, but stable enough to be worn a couple of times on the other hand. Stability over a long time or heavy use, e. g. washing, were not in our focus, because children and young people will typically build prototypes only. We describe the results of our experiences later in relation to the corresponding prototypes.

Some yarns also allow soldering as a connection to electronics. For more details, see table 6.2. Another technology used for the connection is gluing with conductive glue. The textile research institute called Titv[13] used the method when working on prototypes for a textile PCB. We describe the prototype later in the textile PCB section 6.3.2.

In the following part, we introduce our results from testing conductive yarns as basic materials for connections of the different parts of the construction kit.

Yarns

The basic material to work with in active smart textiles, is conductive yarn. We worked with different yarns during our research. The following table (cf. table 6.2) summarizes resistances we measured by using a standard Voltcraft multimeter, namely the model VC140. The yarn usually consists of a number of single threads spun or twined together. The identifier for the yarn can be read in the following way: The number before the x stands for the number of single threads used. For example, 2x275 stands for two threads. Titv assured that they could produce a yarn that has only 4 Ohms resistance on one meter,

Table 6.2: Yarns and resistances

Name	Resistance
L110 34 x u8 Shieldex	236 Ohm
235f34 dta 2Ply Shieldex	87,5 Ohm
253 1x4 Shieldex	41,1 Ohm
Lamé Lifesaver	76,3 Ohm
Copper wire weave width 14 cm	0,4 Ohm
Elitex elastic loose	24,3 Ohm
Elitex elastic stretched	14,3 Ohm
Elitex 235 f 34 PA AG	25,4 Ohm
Elitex isolated (coated)	37,1 Ohm
Bekinox VN 12/2x275	5 Ohm
Bekinox VN 12/8x275,100s,HT	1,4 Ohm
Bekinox VN 14/1x90, 90Z	21 Ohm
Statex Flies, small	0,8 Ohm
Statex Flies, wide	1,7 Ohm
Statex Fabric, wide	2 Ohm
Statex Fabric, thin	0,8 Ohm
Karl Grimm High, 3981-7x3	2,9 Ohm

[13] http://www.titv-greiz.de/

but all the samples we measured had much greater values.

The yarns differ in conductivity, durability, elasticity and ability to be sewn as well as machine processed. For different parts of the EduWear construction kit, we have different requirements. For the weaves and knits, (e. g. for prefabricated connectors that we describe in section 6.3.2), the possibility of the yarn to be processed in the machine is more important than for the yarn children and young people use in the workshops or construction kit to sew themselves. In the workshop scenario, the sew-ability is the most important characteristic as well as a durability during the process and the handling for the young people, because their sewing technique is often not that advanced. In the following part, we introduce the results of our experiences with different yarns in workshops with children and young people in more detail.

Shieldex. The Shieldex yarns[14] are produced on silver basis and their original purpose is isolation and anti static, e. g. in medical applications. The 110 yarn is easy to sew, but high in resistance. It can be used for simple sensors in workshops. The thinner yarns can't be used in combination with a sew-on PCB and knotting. The knots will loose themselves.

The isolated Shieldex yarns are good, because they avoid short cuts, but they cannot be processed in standard sewing machines. We tested them on a Pfaff Smart 100 S in the lower bobbin and weren't able to sew. De-isolation of the yarns when wanting to connect to electronics can also be quite difficult. The isolating layer in some cases has to be removed by heating the yarn and the yarn will start to burn when it is overheating. This technique is not suitable for children and young people to do it on their own.

Karl Grimm Yarns. Karl Grimm yarns[15] are zinc-plated and were originally developed for flowering borders. It is possible to solder the yarns, because of the zinc-plated surface. They are also good as heating yarn, because they can be heated with low voltage (1.5 V in our test set). The problem of working with Grimm's yarns is that they loose their conductivity very fast (the zinc surface will chip) and tend to fray out.

Lamé Lifesaver. Lamé Lifesaver yarn[16] is a yarn that was made to repair lamé for fencing. It has a high resistance, but is very durable. One can process it manually, but also with a sewing machine, quite well. In combination with more precise sensors and long distances on a garment (e. g. an accelerometer on the arm), the values are corrupted, because of the yarn's high resistance. An advantage of the Lamé Lifesaver yarn is the availably even in small amounts. For construction kits, this is an important factor.

[14] http://www.shieldextrading.net/
[15] http://www.karl-grimm.com/
[16] http://members.shaw.ca/ubik/thread/

Bekinox. Bekaert[17] produces yarns based on steel or with parts of steel under the name Bekinox. The thicker ones are too thick to sew, but are good for weaving or knitting. The thinner ones can be sewn, but they tend to fray out.

Elitex. Elitex is a brand by Titv who use the Shieldex yarns and improve their conductivity by further processing. Because they have more steps in production, the yarns are more expensive, but their conductivity is better. The yarns can be sewn by hand or machine.

Conclusion

To work with sensors that require little precision, we conclude that the Lamé Lifesaver yarns are suitable. Otherwise, the best results for the kit were achieved with Bekinox and Elitex yarns when we are using yarns as connection without further processing for the children and young people to sew themselves.

In our kit, there are also prefabricated connectors and conductive textiles serve as connections between sensors, actuators and controller as well as pure textile sensors.

Prefabricated Connectors

The connectors are simple conductive yarns or isolated yarns to sew oneself, as we discussed in the previous section. For EduWear, we also worked with prefabricated connectors available on the market and developed some ourselves. Examples for prefabricated connectors on the market are non-wovens to glue, produced by Shieldex. In prototypes, the glued surfaces save a lot of time compared to sewing. The glued connection is less durable than the sewed one and that can make debugging hard as errors due to broken connections are difficult to find.

As alternative, we developed prefabricated pieces of a weave or knit and patches of fabric with conductive yarn already sewn on. We call this textile “cables” *data buses* (see figure 6.3 as an example). Depending on the connections to the electronics or other textile parts, the buses have to be more or less precise. For a snap-button connection, for example, a huge amount of precision is required. The shrink rate of the weave is high, so the fabrics have to pre-washed before further processing to ensure precise gaps. When children and young people sew onto the connectors themselves, precision is not a critical point.

The Arduino architecture requires to connect e. g. sensors in a certain way to keep a great flexibility. For children and young people, it is not easy to understand and to connect the circuits correctly, e. g. to connect the ground of a sensor patch to the ground of the micro-controller. To ease the connections between the different parts, the woven and knitted connectors can be color-coded. As color code we use “traditional” colors in electronics as basic. Red represents power, black stands for the ground, green represents digital inputs and outputs, and orange stands for analogue connections.

[17] http://www.bekaert.com

Sensors

For the construction kit, we prepared sensors (as well as actuators) as patches. A patch contains the circuit and the electronic components; the user only needs to attach the marked or color-coded outputs to a connector or to the micro-controller.

As (entirely) textile sensors we used a textile switch and a flex sensor in the EduWear construction kit. The latest version of the flex sensor was knitted in a knitting machine with elastic conductive yarn. Earlier version used regular (non-elastic) conductive yarn. The elastic yarn improved the performance of the flex sensor, but it is still not comparable with a standard electronic flex sensor.

For further sensors and switches, e. g. temperature sensor, light sensor or tilt switch, we combined electronic parts with textile patches.

We also used smart materials namely UV-sensitive and glow-in-the-dark yarns. These do not serve as sensors directly, but as a basic material to give children and young people experiences what smart materials are able to do. We produced gloves out of the yarns end embedded conductive lines to make them act as sensors. The thumb is the ground line; each finger is connected to a digital pin. That is how we can use such a glove for very basic gesture detection.

We differentiated between analogue and digital sensors which is important for how to connect them to the micro-controller. A digital sensor will be connected to ground and a digital pin while an analogue sensor goes to ground, power and an analogue pin.

Actuators

Pure textile actuators could be thermochromic materials. We worked with thermochromic fabrics, but it is not easy to provide the material in a construction kit. To make the thermochromic material change its color, the fabric has to be heated. The method to heat the fabric is resistive heating. Conductive yarns with high resistance are placed in the fabric and heated by putting current through. When heating the yarn, one has to be careful not to burn the fabric (by overheating it). That is why we did not consider thermochromic materials to be part of the construction kit. The same argument applies to other smart materials, for example shape memory alloys. In the construction kit, we mostly used standard electronic components as actuators: LEDs, vibration motors and buzzers. The LEDs were also prepared as patches, while the children and young people used motors and buzzers unprepared.

First Prototype

In the first version of the EduWear construction kit, we changed the standard Arduino board layout and developed a three-piece board (see figure 6.3). The three pieces were programmer, controller and power supply. The aim was to wear the least possible hardware on the body. The LilyPad—that did not exist at this point of time yet—uses a similar architecture. To save costs, we relied on serial communication instead of using the FTDI232R chip. Most current notebooks would need an USB-serial adapter, because

they do not have serial interfaces on board anymore, but especially older computers at schools could use their COM interface to connect the programmer. The programmer board is only needed to upload programs to the controller and the power supply board is only needed when working with power sources that supply more than 6 volts. The first experiences with the three-piece layout showed that such a design is error-prone. Mistakes can be made when connecting the three pieces, even though we used standard D-SUB plugs to ease the connection.

Figure 6.3: First prototype. The figurative fabric patches on the left top hide electronic components and can be attached to the woven textile bus underneath with snap buttons. On the right top, you can see the three-piece programmer, controller and power supply hardware prototypes.

For the first prototype, we used luster joints and put in yarn and cable to connect yarns and fabric to the PCB. This solution works well, but does not look good and uses a lot of space. In electronics we have different similar solutions, e. g. plugs in different sizes or clips. For prototyping, we also used crocodile clips, but it is tricky to make stable connections without short cuts. It is also possible to crimp sleeves to the yarn and then connect to electronic parts e. g. by soldering. As the work must be carried out precisely, it is rather a technique to prepare parts of the construction kit, the young users can't do it by themselves. The clips or crimps have the problem that the loose ends of the yarn tend to produce short circuits as they might touch each other.

Second Prototype

For the hardware part, we went back to a solution that is made of fewer pieces. We worked with standard Diecimilla boards, but also created a board layout in a size comparable with Diecimilla, but with holes on the board to sew on (see figure 6.4). To ease the sewing, one can prepare holes on the board where children and young people can sew through. Smaller holes allow smaller PCBs, but make sewing without shortcuts harder. Bigger holes are easier to sew into, but make the entire PCB larger. Sewing, if done properly, ends up with a stable connection. When the sewing was done loosely, the

Figure 6.4: Second prototype from left to right: PCB optimized to sew on with conductive yarns, patches in canvas-style and a patch in a close up—the components are fixed by sewing.

connections were less reliable. To fix the connections, we then used conductive silver. A more expensive solution with a similar result is to fixate the connection with a drop of conductive glue.

Buechley's LilyPad uses this solution with relatively small holes. In our experiences in workshops with sewable solutions, we found the connection between PCB and yarn difficult, as it not only has to be very tight, but also has to be sewn back a long way or knotted precisely. Children and young people have difficulties with that. In our workshops, we also observed that children and young people need to try out things a lot, and it is frustrating if they spend a lot of work and then they have to undo everything. Also, for prototyping, sewing is a lot of effort to try if a connection really works. The LilyPad has a power supply that is not on the main board, but, as in our first version, a different part. We experienced a lot of shortcuts between power and ground line while connecting to the power supply. That is why we decided to put the power supply back on the board. We were using button cells here that are relatively expensive. Also, button cells are not rechargeable, that is why they are no good solution for a power supply.

In the early versions, we knotted and soldered in the electronic parts after bending the legs of the component. The connections were easy to break and labor intensive to make. In the next step, we tried to weave in electronic parts directly on the hand-weaving machine by always stopping the machine, putting in the component, and continue weaving (see figure 6.5 for examples of both techniques). When working on the hand-weaving machine, one can directly color-code with different colored yarns. The problem of the technique was that the weave is too loose to provide a stable connection and the electronic components had to be fixated with solder. In the later versions of the patches, the components were laid out and fixed by sewing. The connection is much better than in the weaving machine, still, when using the patches in workshops with children and young people some of them broke. When one pulls on a leg of the electronic component or the conductive yarn connection, it is possible to pull out the hole thing (as you maybe can imagine when looking at figure 6.4).

Third Prototype

In "traditional" textile techniques we also find connection methods, e. g. zippers, buttons or seams. These connections are familiar to the users, because they are part of their

Figure 6.5: Knotting technique on the left and weaving the components directly into the weave in a color-coded LED fabric on the right.

clothing. Suitable connections in combination with electronics are snap buttons, because they are conductive and connect very tight. Our latest prototype (see figure 6.7 for more details) has snap buttons as connectors. Snap button are produced in different colors so it is possible to color-code the connections on the PCB by using corresponding colors. The downside of snap buttons is that through their tight closing, opening them can destroy the fabric they sit on.

Another important aspect of this iteration, was the change of the patches towards a *do-it-yourself patch*. The do-it-yourself patches did not contain the electronic parts when the user started working with them. They were entirely textile—a color-coded knit—and the children and young people could then stick in the electronic parts themselves. This way the amount of manual labor in the production was reduced and the patches offered great flexibility—e.g. all sensors could be mounted to a sensor patch.

Working with conductive yarns (without isolation)—especially prototyping—was hard, because the yarns had to be placed very neatly. In the third iteration of the kit, a string was introduced to solve the problem. The string covered the conductive yarn with a non-conductive layer (see figure 6.6 and figure 6.7).

Figure 6.6: The third prototype with explanations of the single parts.

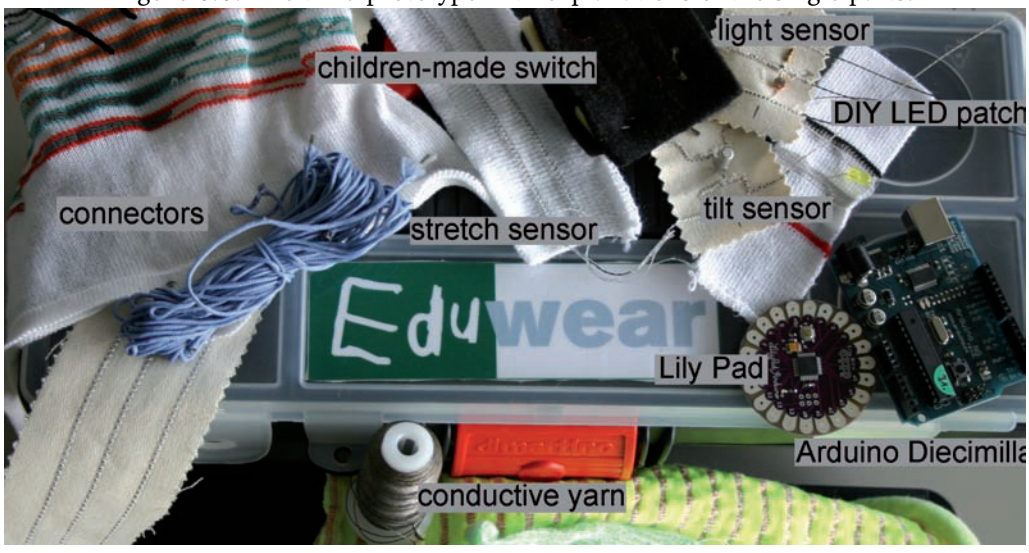
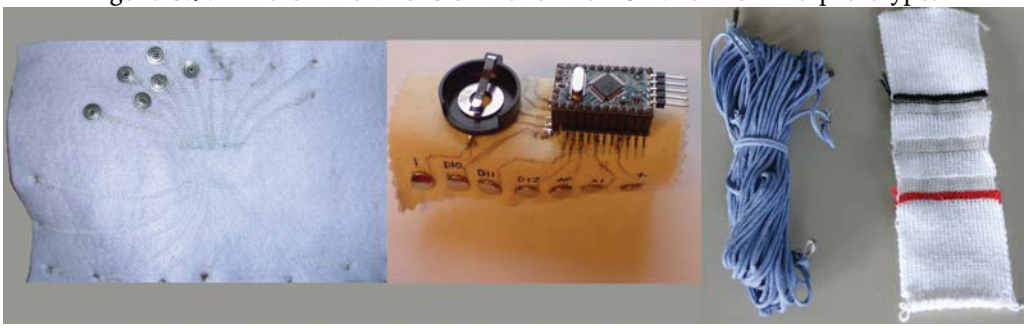


Figure 6.7: Different iterations of the textile PCB and the third prototype.



Textile PCB. We already discussed ways to create fabric or textile PCBs based on embroidery or iron-on or flexible polymer based PCBs in the smart textiles section 5.2. We saw a textile PCB as an important part of a construction kit for smart textiles because of its look and feel that is much more textile and less electronic compared to standard PCBs. In our textile PCB prototypes, we used a method where we embroidered yarn (or rather very thin copper wire, because of the better conductivity) onto the fabric by laying it out and fix it by sewing non-conductive yarn on top. This technique is called *couching*. For the prototypes we did it manually, but Tajima[18] introduced a machine that is able to do couching with conductive yarns or thin wires, according to Ellwanger [Ellwanger, 2007]. We connected the wires to the pins of the Arduino Mini by wrapping the wires around the pins with a wire wrapper—also a labor-intensive process.

To connect the electronic components necessary for the Arduino circuit, we used a method consisting of sticking the components through the fabric and soldering for the prototype. In a discussion about a mass production of the prototype Titv [Reichel, 2007] suggested a method where the components are connected to the textile by sticking them through and fixing the connection with glue (cf. figure 6.8). Titv tested the durability of this method for electronic connections in such a prototype by moving it in 10000 cycles. Our textile PCB ended with color-coded snap buttons to have a connection that is easier to use than a sewable solution. The textile PCB can be connected to buses or sensor patches by snapping them on. That makes prototyping much easier, because children and young people can just snap something together and see if it works.

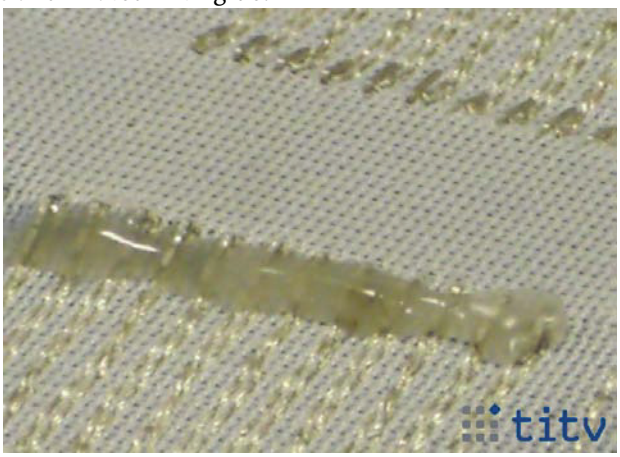
6.3.3 Results from Observing Interactions with the Prototypes

We learned some implications for the design of the EduWear construction kit through observations of the young users with the prototypes. These results are summarized in table 6.3. To produce the patches, we developed a technology based on couching. To improve the durability of the patches, e. g. for a commercial product, we would recommend this technique in combination with conductive glue.

Considering our concept, we can say that familiarities in the design also have their downsides. The figurative patches were familiar, but the associations connected with

[18] http://www.tajima.com/

Figure 6.8: Gluing technique from Titv, with permission from Titv. Components are stuck through the fabric and fixated with glue.



them did not fit into a construction kit. To design them as blanks canvases, allowed a more flexible use. In this interpretation, we would draw on the question of provinces of meaning and assume that the associations to the world of play did not fit into the context of creation. We could also think about the zeug character of the patches—maybe they weren't perceived as an equipment to create with, because they were ready-made and not something unfinished to work with.

The development from one iteration to another can be seen as stepping from familiarities towards flexibility. On the other hand, it is also a development from very work-intensive, crafty solutions towards an automated processing that reduces the amount of manual labor.

Table 6.3: Overview prototypes and main results

Name	Insights	Techniques
Prototype one: Figurative patches and three-piece hardware.	Figurative design is not flexible enough. It did not fit in the children and young people's imaginations and associations they came with to the workshop.	Crafty solutions: Bending electronics and knotting yarns.
Prototype two: Sew-on board and canvas-style patches	Sewing on connections is very hard for children and young people. The connections turn loose after a short while. The canvas patches are more often used.	Experimental: Weaving with laying components into the weave.
Prototype three: Textile PCB with snap on buttons, Do-it-yourself patches, string.	Only partly tested with children and young people. Promising technology for the future. Couching makes connections that have a short durability only.	Couching: Couching can be done with the machine. Do-it-yourself patches can be woven with a machine entirely.

6.3.4 Conclusion

On the conceptual level, we moved from a rather familiar design to a more flexible one and showed how important the adjustment between familiarities and flexibility is. Of course, with the canvas-styles patches we still rely on familiarities—the building block metaphor—that work well.

On the technical level, we introduced a method to create a color-coded smart textiles kit including a textile PCB, textile buses and patches that is easy to use compared to sew-on solutions as it relies on snap buttons. We tried optimizing the process to avoid waste and manual labor, but still fitting in the electronic parts has to be done by hand. The method is not entirely new, but a new combination of existing methods.

We also tested different yarns in our workshops and can provide recommendations which ones to use for a learning environment, in which children and young people that are not highly skilled in textile crafts work with the materials.

6.4 Programming Environment

In this section, we will introduce Amici, our tag-based programming language. The programming environment is based on the Arduino language and is meant to allow the children and young people to make it their own place and integrate their context through tagging. The ideas draw on familiarities and specific usage patterns of children and young people—to use web-based applications as social media. You can download Amici under http://www.dimeb.de/eduwear; it is currently running on Windows and Mac OS X.

As we did in section 6.3, we start by discussing the programming environment's concept and its relation to our design principles before we move on to the technical implementation.

6.4.1 The Concept

We already discussed working in a participatory design process in section 6.1. For Amici, the following design principles were most important to keep in mind:

- Start designs from familiarities.
- Create a shared context and situate the learning process in there.
- Create (third) places.
- Allow the communities to create and design their own place.
- Bring contextual information in the environments.
- Make the programming environment a place and draw on familiarities.
- Design for wide walls and for flexibility.
- Carefully adjust between flexibility and familiarity.

We also mentioned "Create a shared context and situate the learning process in there" and related to DiMeB's workshops (see section 6.1.1) with the principle. Now, we try to relate to this context not only by using the software in it, but also through choosing the software's metaphor according to the context.

We introduced social software (see section 5.1 for details) and showed that it is part of the children and young people's life-world. We take up social software, especially tagging, and a suitable metaphor (see section 6.4.1) to draw on familiarities. Tagging also serves as flexible technology to create and design the programming environment (see section 6.4.1 for our hypotheses about tagging in Amici).

The Metaphor

On the software side, we use a computer clubhouse as we know it from the real-world (for example from the workshops we organize in the research group DiMeB and we described in section 6.1.1), as metaphor. Such a clubhouse is a *familiar third place* for

our target audience. To place the learning process in a context is one of our design principles that is why we consider the metaphor as a fitting one. The clubhouse has different tables where children and young people sit and work on their projects usually with physical materials as well as with a notebook. Children and young people can walk around to look at the others' projects and talk to the other children and young people or to a tutor. Such an environment is a dynamic complex system, materials, setting, pedagogic concepts and likewise influence the situation and the learning process.

Let us illustrate an example situation we are aiming at with a revised scenario based on our vignette from the beginning.

> *Lena took her favorite T-shirt to the workshop in order to make it display her mood in light patterns. This way her friends know, if she feels like having a chat or rather be left alone. She sews in a nice design with prepared LED patches and temperature sensors (to "measure her mood"). As she boots up the programming software and logs in, some project ideas are displayed. Last week Lena saw Gini working on a nice program, but how did she program it?*
> *Unfortunately, Gini is not around this week. Lena opens Gini's project—a bag that lights up in the dark—, because she likes Gini and her project and takes a look at Gini's programming. Afterwards, she tries herself to program on a visual basic. When she moves her mouse over elements some tips are displayed. Still, she can't figure out how to work with the "if". So, she clicks on the help button. Some example projects are displayed that are similar to Lena's own project. "There! Temperature sensor. That is what I want." Lena takes a look at the program and goes back to her own project knowing how to go on.*

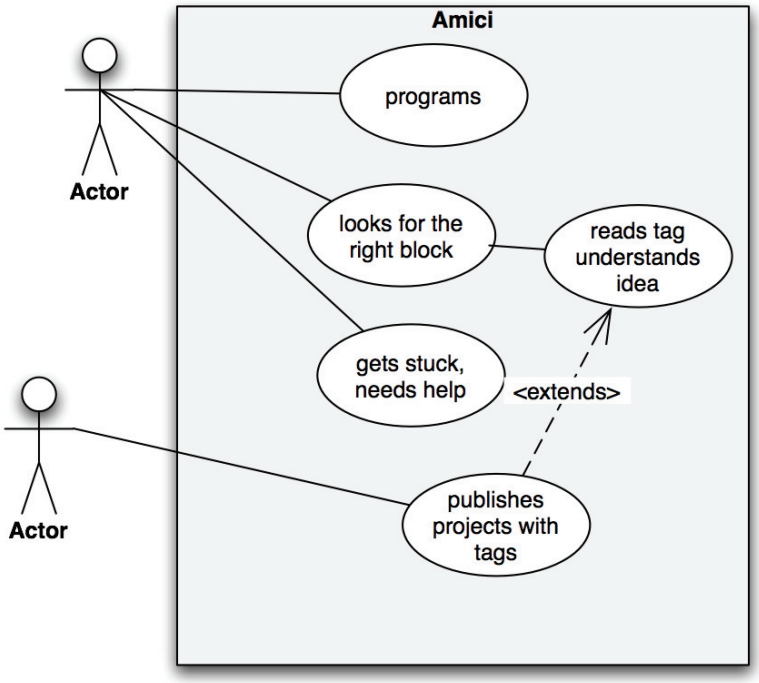
This scenario formalized as a use case looks like figure 6.9. By using the computer clubhouse as metaphor, we draw on a third place metaphor. The children can look around (in the software they can browse through users and their projects) and ask other users for support or simply for a chat out of fun. They can also call blocks how they think they should be called (to design their own programming place), through tagging. A user might use a block to make her T-shirt blink. To remember how she used the block, and to tell the other users how the block could be used, she could call it "blink" with a tag. To facilitate browsing and also finding users that might know how to solve a problem or that might be interested in a chat, because they have similar interest, users have a user profile, but also use tags to call blocks and to annotate their projects.

Hypotheses: Tagging

On the conceptual level we use tagging to support multiple goals. We assume the following:

1. Tagging is a way to make programming and support easier and context-dependent. To allow the young users to integrate their context and embed the context in the software, we need a way to attach meta information—parts of their context. Due to

Figure 6.9: Simple use case of the Amici software.



the ease of use, we try to encourage children and young people to integrate contextual information through tagging and share their projects based on the descriptions and their ideas what blocks stand for through tags. We described this function of tagging in the metaphor section 6.4.1.

2. Tagging is a way to influence the design process the children and young people go through, because their field of possibilities widens through seeing different tags related to one block. Tagging is flexible enough to be used to describe what somebody associates with a block on several levels. We also argue that seeing other users' projects will change the designs the children and young people want to work on. They might find a great idea that is an inspiration and causes them to add new functionalities to their design. Seeing a similar project, might also make the users give up their ideas, because they found out other people already did something similar.

3. Tagging can draw on familiarities and positive associations. In our concept, we referred to the importance of familiarities and in chapter 5 we pointed out that social software and tagging are familiar to the children and young people. We aim at a positive impression and an association to fun and social networking sites by adapting tagging in Amici. To use Schütz' terminology: We assume that tagging is a part of the life-world of the children and, furthermore, it is part of the world of leisure and friendship.

4. Tagging can help to support multiple interpretations. In section 2.2.1, we pointed out that everybody will interpret a tag differently—depending on the context. Experiences and associations play an important role, but also the intention, what the children and young people want to use a block for. A tag becomes a zeug in the sense of Heidegger. A designer will interpret a tag on the block level depending on what she wants to program, what she wants to achieve by using the tag.

5. Tagging is a way to design and appropriate a (third) place. Instead of dealing with predefined categories that depend only on the programming environment's designer's world view, children and young people make the programming environment themselves, as their place. With tagging, the users construct the environment.

6. Tagging can be used as a tool to trigger reflection and memory. When tagging, for example, a project, the tagger has to reflect on what the project actually is and what it does. Seeing self-constructed tags on a block can help to remember, how one used the block before.

6.4.2 Technical Details and Implementation

Visual Programming

Our software is starting with a dialog to allow the users to login. When a name is entered that is not yet in the database, the account will be created. When the user logs in, a screen with a faraway look on users and their projects is displayed. Just as in the real computer clubhouse children and young people that are using the software can look around and gain an overview about what is happening in the system. When a user finds a project or another user interesting, she can click on it and get a more detailed look. We use a *zoom* metaphor here. The user can zoom into the project and open it. The project then will open in a pop-up window and a picture of the project, title, tags, source code and the user who created it are shown. The project can be opened as block code from the pop-up (the different zoom levels can be seen in figure 6.10, figure 6.11, figure 6.12 and figure 6.13). On the next zoom level, the project is represented as a program in the visual programming language we developed. Children and young people can drag and drop blocks that represent a command on the canvas. The different blocks are displayed in a frame on the left. On the right is a blank canvas (cf. figure 6.14).

Each block stands for one or more method calls in Arduino. The users drag blocks to the canvas, a twenty-five times twenty-five grid, where at the beginning a setup and a loop block sit. All blocks right below the setup, the loop, or a user-generated method block, are interpreted.

When changing to the code view, or uploading the program, the blocks are translated into Arduino code.

The block metaphor we use is a similar model to the one Begel presented in 1996 [Begel, 1996]. As important changes, we save in a human-readable XML format opposed to binary formats and integrate search and publishing in the programming environment as well as to allow adding metadata to the objects in the environment.

Figure 6.10: Zoom level one: the users in the system. The center node is the user that just logged in; the other nodes are centered around her, the distance depends on the similarity.

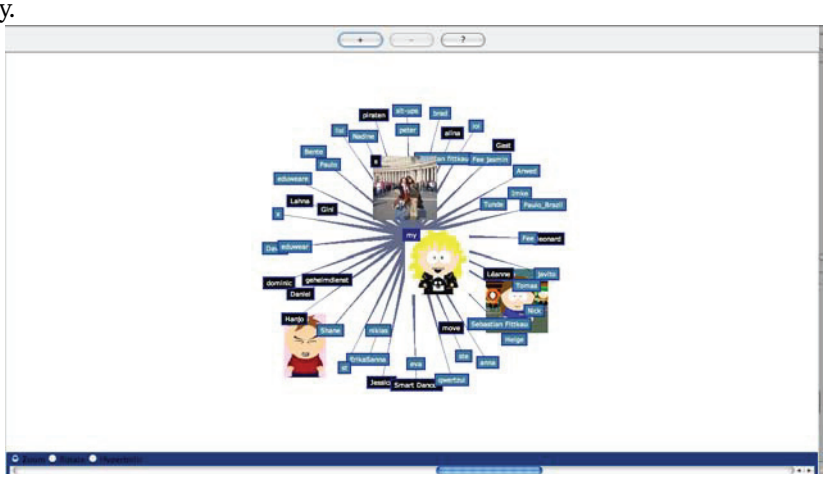
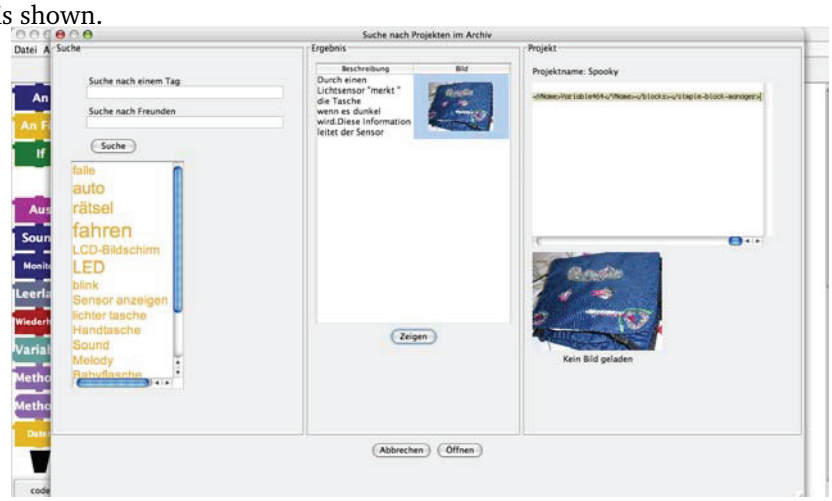


Figure 6.11: Zoom level two: details of a project. On the left, you can see the search menu and the system's tagcloud. In the middle, you can see the project with its description. On the right, a larger version of the project's image and the source code in XML format is shown.



Each block contains a connector that indicates whether or not another block on the right is required. The connectors are drawn dynamically, depending on their position on the grid. For example, the "If" block needs a block on its right to determine the action executed when the condition is true. The blocks can be dragged below the "Loop" block that is on the canvas, when the program is started, or under the "Setup" block. Blocks under "Loop" will be executed in an indefinite loop, while the "Setup" method will be executed once, after the program starts and before it runs into the indefinite loop. This structure is part of the Arduino programming language and we kept it, because we experienced problems when teaching e. g. Cricket Logo to children and young people where methods to retrieve values from a sensor have to be surrounded by a loop construct (see the following listing).

Figure 6.12: Zoom level three: code in the visual representation. On the left, you see the blocks the user can choose from, in the middle, a block labeled “Loop” and, underneath, a block labeled “On” followed by an if block connected to a block labeled “Sound”. The program will run into an indefinite loop and always “switch on” pin 2. Afterwards, it will ask for the values from pin 0 and in case the value is less than 45 play a sound.

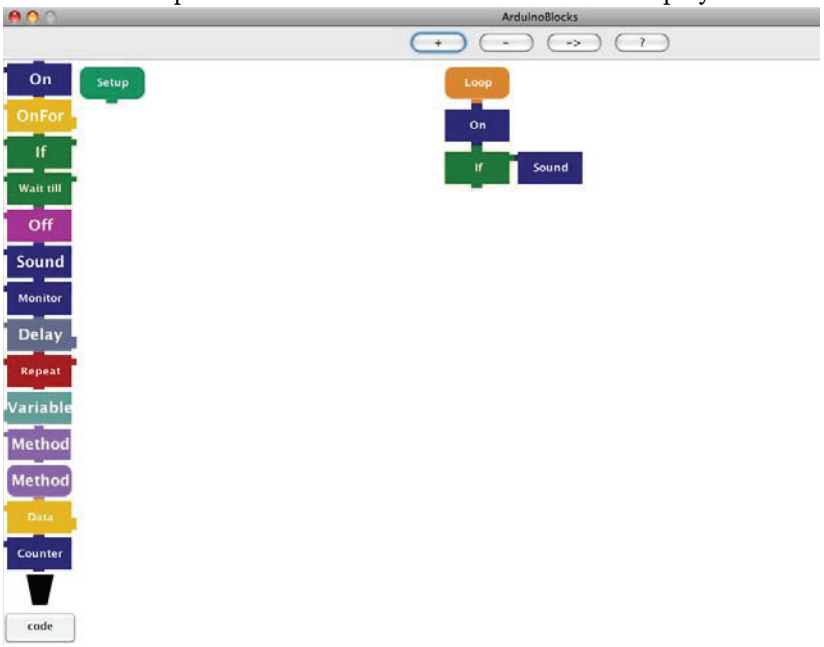
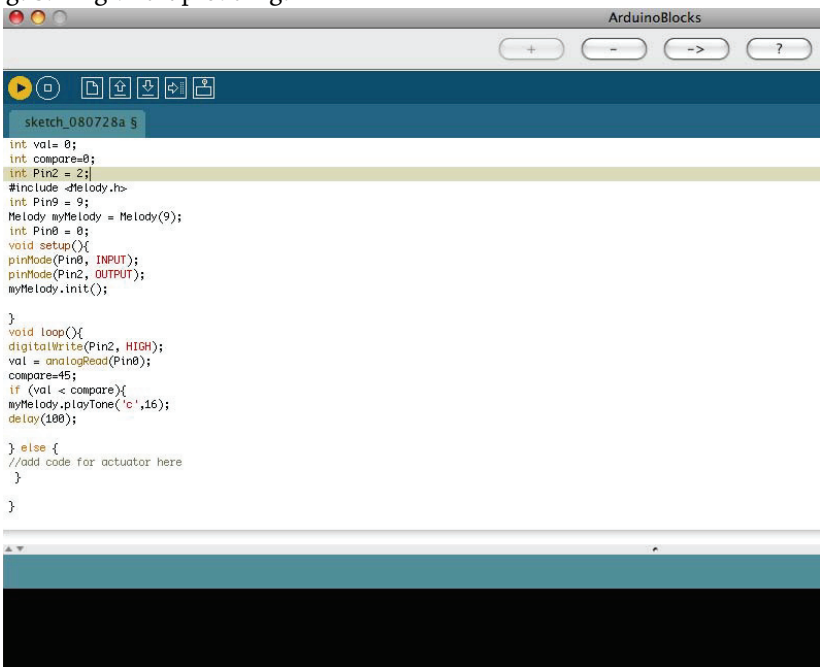


Figure 6.13: Zoom level four: the standard Arduino code and the functionality Arduino offers, e. g. saving and uploading.



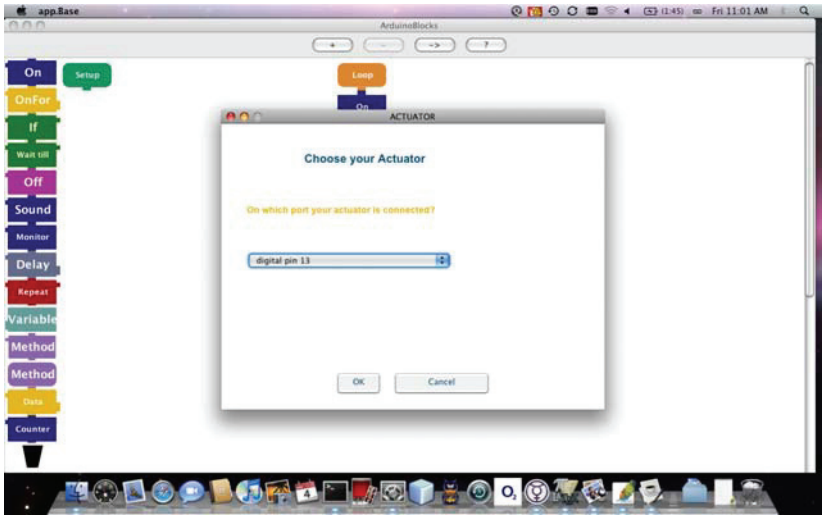
Listing 6.1: Retrieving sensor values with Cricket Logo

```
1 loop [
```

```
2     if sensora < 125 [a, on]
3   ]
```

In Amici, users can also create their own methods by dragging a “Method” block somewhere on the canvas and the desired blocks underneath. To switch on an LED attached to pin 13 on the board, one needs to drag an “On” block below the “Loop” block. After the drop action, a dialog will appear to allow the user to add information, e. g. what pin the user wants to switch on. To go back to the example: One would select pin 13 in the dialog (figure 6.14 depicts the dialog). The program would be complete and could be uploaded to the hardware. You can take a look at some example projects in the appendix A for further explanations on how to program with Amici. Technically, we

Figure 6.14: The user switches an LED on. After dragging the “On” block to the canvas a dialog box asks for details on the connection.



differentiated between blocks of different kind: We have the class *ActionBlock* and the class *Block*. ActionBlocks are inherited from Blocks. The action blocks are all blocks that represent functionalities where an actuator is controlled. Examples are “On”, “OnFor” or “Sound”. The details of the class structure are shown in the class diagram 6.15.

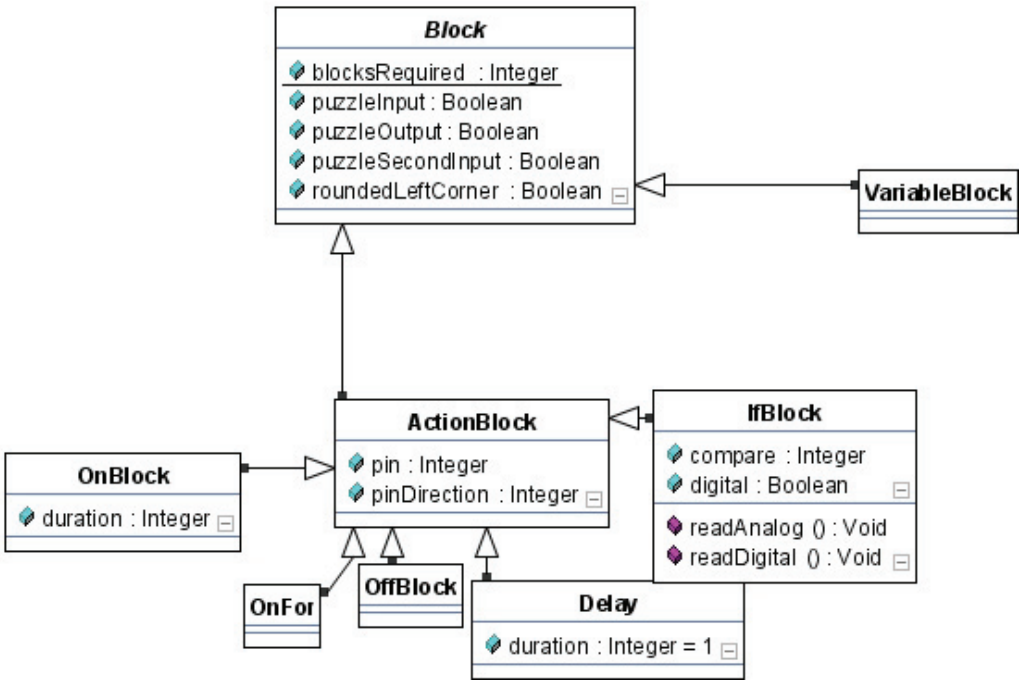
Technical Basics: The Arduino Software Architecture

By zooming even further into the open project, the user can see the program in its representation in high-level source code. In our case, this is again Arduino, a domain specific language based on C++ to program the Arduino board. The zooming only works in one direction—we can make code out of blocks, but not blocks out of code. The easy operation from the visual programming layer (see figure 6.14) would look like the following code example in the written source code.

Listing 6.2: Arduino basic elements

```
1 int ledPin = 13;
2
```

Figure 6.15: Block architecture. All blocks are part of the class “Block”. Blocks that use output pins are “ActionBlocks”.




```
3  void setup()
4          {
5           pinMode(ledPin, OUTPUT);
6          }
7  void loop()
8          {
9          digitalWrite(ledPin, HIGH);
10         }
```

The code is much more complicated than the representation in the visual layer. On the other hand, it reveals more of what is actually happening when the user wants to do something as easy as switching an LED on. First, the pin on the hardware has to be declared as an output, it could serve as well as a digital input and one could connect e. g. a button to it and read in values of the button. After declaring the pin as an output, it is possible to “switch on the pin” by setting it to “high”.

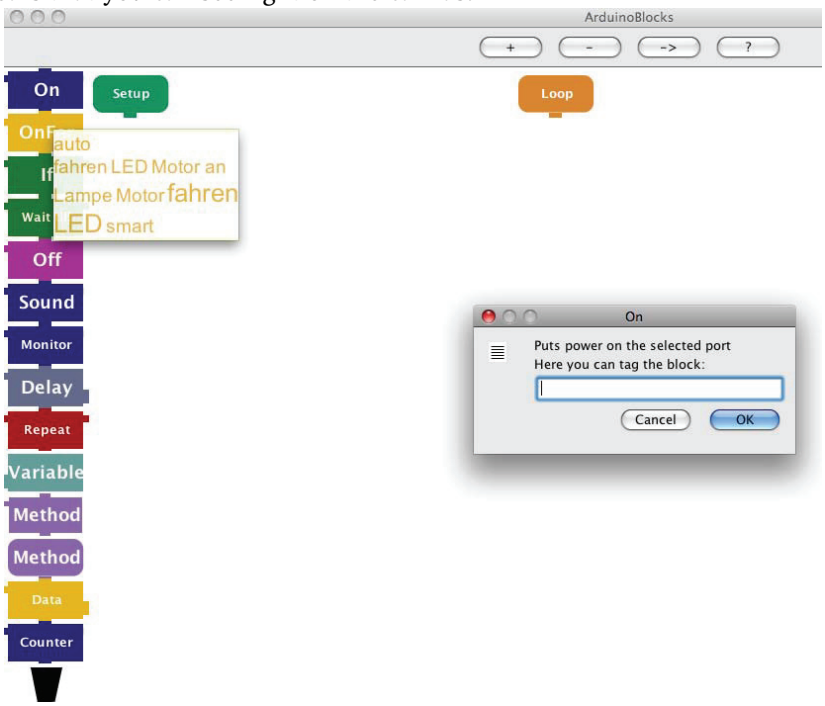
We considered hiding more of what is happening and represent the visual block code in another layer, an easier Logo-like language. In first test sessions, we discovered the children and young people programmed in the visual layer and then read the source code and understood most parts of it. We consider this as a great learning process. Therefore, we decided to use “complicated” code, the users can still stay with the easier visual programming and get into real understanding through engaging in the source code.

Technically, Arduino itself is a C and C++ library based on Wiring. The programming environment is Processing that is written in Java. The Arduino environment uses an avr-gcc compiler and AVRlibc to compile the Arduino code to an Atmel compatible C, avrdude and UISP to upload the code to the hardware. The software relies on Java 1.4. to be able to ship the Windows version with a Java runtime. Arduino structures its programs (called sketches in the Processing style) in a *setup* method that is executed after the bootloader ran and a *loop* method. The loop method is an infinite loop that starts to run after the setup method. We already mentioned the model—that we can find in all members of the Processing family—with the visual programming layer.

Tagging: The Implementation

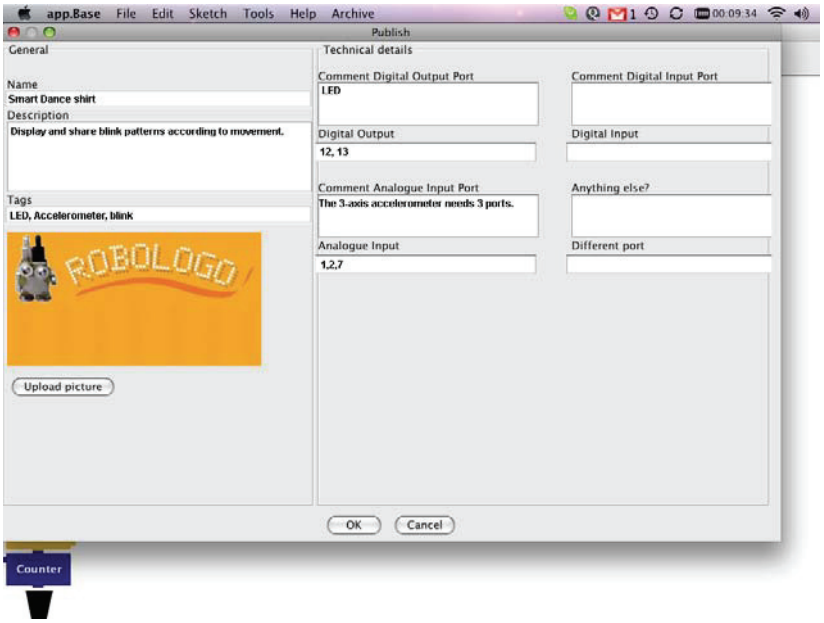
We mentioned that tagging can be done at the project level and at the block level and now we explain in detail how we implemented these different levels of tagging. On the block level, children and young people can right-click on one of the blocks in the frame on the left and just add a tag. The tags on the block level are visualized through the "common" tagcloud as a tool tip. This means, the tagcloud will appear when the user moves the mouse cursor over a block on the block frame. Within the tagcloud the occurrence of the tag is visualized by the font size in which it is displayed. The children and young people can tag the blocks with whatever tags they find appropriate (see figure 6.16).

Figure 6.16: Tagging on the block level. In the left corner, you can see the tagcloud for the "On" block that appears as tooltip on a mouseover. After a right-click, the tag dialog box appears that you can see right on the canvas.



The second level of tagging in Amici is tagging on the project level. When a child finished a project or wants to save the current work, she can choose *publish* from the menu panel. A dialog appears in which she can upload a picture of her project; give the project a title, a description, tags, and some additional information. The additional information can be entered into a form and refer to what kind of sensors and actuators are connected on which pins on the hardware. While the tags are free-text, the information about pins should be more formal for further processing (the dialog is depicted in figure 6.17). The

Figure 6.17: Publish and tag a project. The dialog appears in the "Archive". Users can describe and tag their project and attach images (as well as attach the code automatically).



projects that consist of this information are saved within the database on the web. From there on, it is further processed to calculate a similarity between users and projects based on the tags and additional information. We discuss the algorithm in section 6.4.2. The result is visualized with the open source library TouchGraph and depending on the view only a subset (the most similar) of users/projects is displayed which we explain in detail in section 6.4.2. Their similarity to the user is displayed in the closeness of their vertices in the graph.

Users can rely on the visualization, but also choose the *archive* and search for projects by keywords or by using the tagcloud that is displayed in the search dialog. Other clients, e. g. the web-client we introduce in section 6.4.2, use the database and display the projects as information.

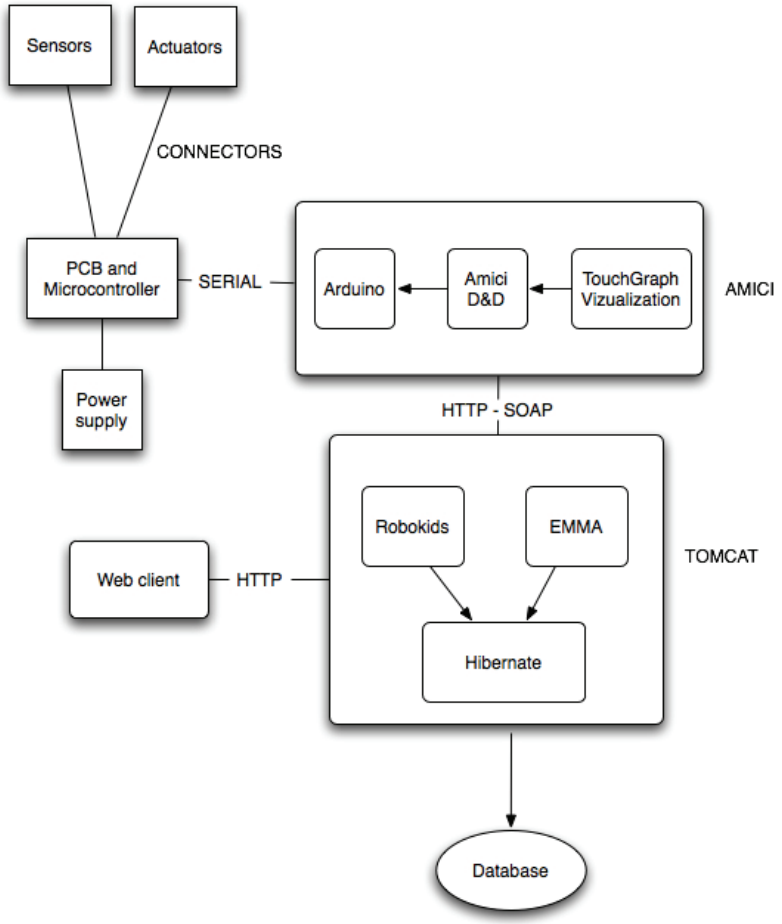
Overall Architecture and Technical Details

We already named the different parts of the EduWear construction kit in section 6.3. These are again: a hardware part that includes a micro-controller, a way to program the micro-controller (the serial or USB connection) and a power unit, sensors, actuators,

connections between the different parts and a programming environment. Hardware, sensors, actuators and connectors are either (smart) textiles or made to be attached to textiles easily.

The software architecture of Amici also has different layers as you can see in figure 6.18. The core layer is Amici: the Arduino layer to compile Arduino code and upload it to the micro-controller, the visual programming language on top and a layer to visualize, create and manage projects and users. This layer is connected to a Tomcat that runs two webservices, namely Robokids—for the basic web-based management of data—and EMMA as recommendation engine. Both webservices are introduced in more details in the next sections. As further client to display projects you can see the web-based or browser-based client, our *virtual lab*, that is directly connected to the Tomcat. We

Figure 6.18: Overall architecture of the EduWear construction kit. The core is Amici, the visual layer on top of Arduino. Amici talks to the webservices, on the one hand, and to the micro-controller, on the other hand.



mentioned Arduino's software architecture, in which the environment is written in Java. Therefore, we decided to use Java as our basic language. Java is also platform independent, an important aspect, because Amici was used from all partners of the EduWear

project, and some partners, for example, used Apple computers. To be able to ship Amici with a runtime, we used Java 1.4 instead of a more current version—we kept the same runtime Arduino uses. The visual programming layer uses Java Swing and Java's Drag and Drop library. To save projects in XML, we use the library Castor[19]. The visualization of projects and users on the client side is implemented with TouchGraph[20] respectively Adobe Flex[21].

The webservice is implemented using the Axis Framework[22] by Apache[23] and runs within the application server Tomcat[24]. As communication protocol between webservices and clients, we work with SOAP that is based on XML and comes with the Axis framework.

Access to the database is implemented with Hibernate[25], a persistence layer for databases that allows to directly save and retrieve objects in their current state to the database.

For logging, we use Commons Logging from Apache and Log4j[26]. The webservices currently run on a server using Ubuntu[27] as operating system, on a Apache webserver and a MySQL[28] database.

Robokids

As webservice for the basic operations we worked with Plassmeier who implemented the Robokids framework in his diploma thesis [Plassmeier, 2007]. We took up Robokids and added a few additions. Different clients can implement Robokids and use the methods the webservice offers. The functionality of Robokids includes:

1. save projects
2. add tags to projects
3. comment projects
4. login
5. get tagcloud for projects
6. get user details
7. save user details
8. get tags of a user

[19] http://www.castor.org/
[20] http://sourceforge.net/projects/touchgraph
[21] http://www.adobe.com/de/products/flex/
[22] http://ws.apache.org/axis/
[23] http://ws.apache.org/
[24] http://tomcat.apache.org/
[25] http://www.hibernate.org/
[26] http://logging.apache.org/log4j/
[27] http://www.ubuntu.com/
[28] http://www.mysql.de/

To save and communicate with the webservice, we have two different entities: projects and users. These are wrapped in transfer objects, e. g. the TagCodeWrapper that wraps a project. A project consists of the code (represented in blocks saved as an XML format), a user who created the project, title, description, tags and photo. There is a free text description, but also a more formal description of the project that asks the user in a form for used pins, sensors and actuators. The values entered in the form are saved as a hashtable and appear as the field techDetails in the project. The user has a name, description, email, tags, image and friends. Tags can be tags for projects or tags for blocks.

Tag-based Recommendations

We have seen in section 5.1.3 and in section 5.1.5 that tagging systems often grow large. To display all e. g. tags of huge systems is impossible and users can be easily overloaded with information. We introduced research on improving retrieval and search in tagging systems in section 5.1.3 to address this problem. For our tag-based programming, we have to limit the number of tags and projects shown. Also, as our young test persons remarked, other projects can only be a help when they are really similar. We limit the number of projects and users that we show therefore and only show similar items—calculated with a recommender that is part of Amici.

We rely on cooperative or social filtering opposed to content-based filtering. We base Amici's recommender engine on EMMA, Gehrken's enhancement of Begelman et al.'s tag cluster algorithm that we already introduced in section 5.1.3 (cf. [Begelman et al., 2006] [Gehrke, 2008]). Begelman et al.'s algorithm relied on tags alone and could detect similar tags quite well. However, as projects our users build with the EduWear construction kit are also very different on the hardware side a tag-based recommendation alone might not be helpful. For example, there might be different projects tagged with "LED" and "bag". On a conceptual level, these projects are quite similar: They are bags that emit light in certain cases. On a technical level, they can be entirely different. While one of the bags may switch on an LED on pin 13 when the light sensor on pin A0 reaches a certain level, the other bag could switch on a row of LEDs on pin 2 to pin 8 when the switch on pin 13 is pressed.

When running one of the project's code on the other project's hardware (the bag) it won't work at all as the hardware is connected differently. The tag-based recommendation would feel similar at first sight, and the child would probably try to run it on her own project. This would naturally fail and we would confront the children and young people with the same situation that we described in our vignette. In EMMA, this problem is addressed by using the information on hardware, e. g. what pin is used for what sensor or actuator, as further information. Technically, they are added to the pool of tags the projects have. MOTORCONNECTIONTAG, for example, is a field the user tags with a number of a pin used for output. SENSORCONNECTIONTAG is a tag with the number of an analogue input pin. In the algorithm that is the step of "add description tags".

The algorithm goes through the following steps:

1. Add description tags to the tags of the project.
2. Calculate related tags or *co-tags*.
3. Search for related projects.
4. For each project found: add description tags, calculate intersection of tags, calculate similarity, save similarity to the projects, and add to collection.
5. Cluster, order the collection according to similarity and return collection.

The basic assumption of the algorithm is that tags that are used together are similar. In the case of "LED" and "bag" that sounds strange, but if a glowing bag is an idea children and young people built often, these tags have a relation. We have shown in section 5.1.5 that users who are connected e. g. by friendship use the same tags more often than random users so tag usage and connectiveness seem to be related.

Tags that appear together to describe a project are called *related tags* or *co-tags*. In the second step of the algorithm, we look in the database for these co-tags. Number of occurrence of the co-tags is counted and the most common co-tags are labeled as "strongly related". We mentioned the problem that some tags can be very dominant as they are used together with almost everything. In our domain, the tag "LED" could be such a tag. To reduce the influence of the tag, the formula to calculate how strong the relation is is important. In section 5.1.3, we already mentioned that Begelman et al. recommended *dice similarity*.

For the recommendations, objects with similar tags are filtered and within the set of objects the similarity to the reference project (the current project of the user) is calculated by building the intersection of tags. The similarity value is based on the size of the intersection. An ordered list of objects is returned.

EMMA also allows choosing between different similarity measures. Some are based on sets others are based on vectors. The *overlap* method calculates how much the sets overlap. It returns a value between zero and one.

$$sim_{overlap}(A, B) = \frac{A \cap B}{min(A, B)} \tag{6.1}$$

Cosine similarity regards the set as vectors and calculates their cosine.

$$sim_{cosine}(A, B) = cos(\vec{A}, \vec{B}) = \frac{A \cap B}{\sqrt{A} * \sqrt{B}} \tag{6.2}$$

Jaccard similarity calculates the coefficient.

$$sim_{jaccard}(A, B) = \frac{A \cap B}{A \cup B} \tag{6.3}$$

We already mentioned the dice similarity.

$$sim_{dice}(A, B) = \frac{2 * A \cap B}{A + B} \tag{6.4}$$

As illustration see the following example from Gehrke [Gehrke, 2008, 58]: We have two projects—two sets of tags.

$$A = \{Car, move, blink, LED\} \tag{6.5}$$

$$B = \{Car, move, blink, yellow, Shirt, Bag, Fabric\} \tag{6.6}$$

In the different similarities we reach the following values:

1. $\text{sim}_{\text{overlap}} = 0.43$
2. $\text{sim}_{\text{cosine}} = 0.57$
3. $\text{sim}_{\text{jaccard}} = 0.38$
4. $\text{sim}_{\text{dice}} = 0{,}55$

The result of the similarity calculation is an undirected weighted graph with an edge between tags, when these are strongly related. Using *spectral clustering*, as Begelman et al. suggested, the graph is divided into clusters on edges with a low weight.

The same algorithm can be used to recommend users instead of projects without big modifications.

Virtual Lab: Browser-based Client

As you can see from the system's architecture (cf. section 6.4.2), it is possible to let different clients communicate with the webservice and retrieve tags, users and projects this way. Next to the programming environment Amici, we implemented a browser-based client. The client is written in Adobe Flex and can be accessed in the Internet with a browser. Flex creates websites based on the Flash format. Users that already worked with Amici, can log in here and add projects and data (e. g. edit their user profiles or comment and tag other people's projects) with their accounts. The user accounts—the information about the users—are extended, when comparing to the implementation in Amici. Several additional features are implemented in the webservice as well, e. g. rating of projects, but up to now not implemented in a client. It is also possible to register as a user via the web client and upload or tag projects when one does not have the possibility to run Amici, or has interesting projects written in another language or is generally interested in the projects, but not in making them oneself.

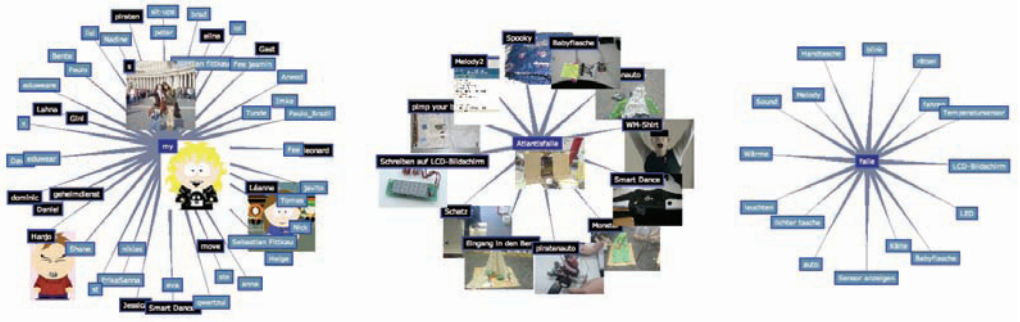
Visualization

To compare view models, we visualized projects, users and tags in different ways. Depending on the view, a graph is built with one of the entities as nodes. In the *user-centric* view, our graph is a weighted graph with the users as nodes, and the similarity between users as weight for the edges. The *project-centric* view has the projects as nodes and the *tag-centric* view tags. We use a graph instead of a tree here, because each node can be

selected as center node; there is no defined root. However, when starting the software, the current user or the current user's last project is displayed as center node.

To implement the visualization of similar users and projects, we used the open source version of Touchgraph[29]. Touchgraph is a graph visualization library for Java and it is used for web-based data visualization. Its layout algorithm is a *spring force* algorithm that handles coalition and is able to interpret weighted graphs. The weight of the edges determines the positions of the nodes in Touchgraph's layout algorithm.

Figure 6.19: Different views from left to right: user-centric, project-centric, tag-centric



6.5 Conclusion and Relation to Constructionism in Context

With the EduWear construction kit and Amici we designed and developed a constructionist learning environment that is based on tagging and smart textiles to implement our concept of constructionism in context. We draw on familiarities (children and young people are familiar with social software and also with using clothes for communication) and on research on how children and young people use social software as well as clothing in their real life. The (smart) textiles material is also a physical material that acts in the world and possesses physical characteristics important in the design process e. g. how the material feels, its softness, stiffness and how it smells and looks. In both, research on social software and in our participatory design process we found indicators that social connectivity is something very important to the children and young people who we wanted to work with regarding the construction kit.

Through working with the textiles, children and young people can connect to their world of everyday life by designing clothes and use technology to add to the design. In our revised scenario, Lena could bring her favorite shirt from home to make it smarter in the workshop—the artifact then would have a personal meaning for her. During the design process, we found out that the material should invite the children and young people to create something with it, to be flexible to fit into different designs and applications.

To put context in the very center of our implementation we did not only take up familiar domains both on the material as well as on the software side. We placed the

[29] http://sourceforge.net/projects/touchgraph

real-world context in the programming environment from the design on—the metaphor we started with is a social one that considers the real-world learning-environment and we designed the (visual) programming language around it.

We also integrated sharing of projects and tagging directly into the programming environment—a social networking site is part of the environment—, because social software is familiar to children and young people and using social connectivity features (as we mentioned before with e. g. YouTube's friendship functionality) is something they like to do. Children and young people can share their project directly out of the programming environment. Tagging is possible on different levels: as tag-based programming or tagging on the block level as well as to add metadata when sharing projects. Through tag-based programming, children and young people can add tags to blocks. When we think back to our vignette, tags e. g. "onlyAcrobot" could have helped to prevent the young programmer's misunderstandings. On the other hand, they could use the tag-based programming for inspiration—getting new ideas on how to use a block.

Tagging on the project level can help to integrate the young learner's context, their ideas about the project they are working on. Because they are connected to the database, they can use other people's projects as examples, inspiration and help system. To support them in this kind of usage, we work with a tag-based recommendation system, so children and young people can find examples from a similar context.

Chapter 7

Evaluation: Piloting and Testing the Prototypes

With EduWear and Amici we aimed at showing that our design principles can help the designer to design a constructionist learning environment. Furthermore, we assumed that through focusing on the user's context the environment would connect to the children and young people's life-world strongly.

When developing the argumentation we hypothesized the following:

1. An interesting object enhances interaction between the learners.
2. Tagging is a way to make support easier and context-dependent.
3. Tagging is a way to influence the design process the children and young people go through, because their field of possibilities widens through seeing different tags related to one block.
4. The EduWear construction kit (especially the smart textiles) relates to the participant's life-world and to popular culture.
5. Amici makes programming fun and a social activity.
6. Tagging can draw on familiarities and positive associations.
7. Tagging can help to support multiple interpretations.
8. Tagging is a way to design and appropriate a (third) place.
9. Tagging can be used as a tool to trigger reflection and memory.

Within the evaluation, we also aimed at finding out what the children and young people find interesting in a project.

In this chapter, we try to confirm our hypotheses to some extent. We use a method triangulation: We combine different methods to approach our hypotheses. In the first part of this chapter, we discuss results from usability testing we carried out concentrating

on Amici. In this part, we want to show that the software is usable for children and young people and we also want to approach the following hypotheses:

- An interesting object enhances interaction between the learners.
- Tagging is a way to make programming and support easier and context-dependent.
- Tagging is a way to influence the design process the children and young people go through, because their field of possibilities widens through seeing different tags related to one block.

We approached the first claim by testing the three different views that you could see in figure 6.19, but we do not discuss the results in detail. Figure 7.3 summarizes the results and makes clear that the children and young people we worked with clearly preferred to see the others' projects instead of the other users' avatars or tags. By giving children and young people a task at hand and measuring task-completion time (for details on the methods see the next section) we approached the second hypothesis. For the third claim, we relied on additional data we collected while the children and young people worked on their tasks. The additional data also helped us to find out what children and young people found interesting in a project.

To approach the following hypotheses, we deployed qualitative methods (see section 7.2):

- The EduWear construction kit (especially the smart textiles) relates to the participant's life-world and to popular culture.
- Amici makes programming fun and a social activity.

We looked at the individual's experiences to see how and why the children and young people worked with and perceived the material.

Concerning the following claims, we also worked with qualitative methods for the same reason:

- Tagging can draw on familiarities and positive associations.
- Tagging can help to support multiple interpretations.
- Tagging is a way to design and appropriate a (third) place.
- Tagging can be used as a tool to trigger reflection and memory.

7.1 Usability

To evaluate if the tags on the blocks—tag-based programming—had an impact for the users, we tested the different possibilities between the test persons. One group worked with tag-based programming, one without. Within the tests each child or young person had a number of tasks to do. These were:

- *Search for a project where LEDs were used. If you like the project, add its maker as a friend.*
- *Look for two interesting people, and add them as friends if you like them.*
- *Program an LED to light up on Pin 13.*

The first half of the group used a regular block view (the blocks showing only the name of the command e. g. "on") while the second half used blocks displaying additional tags (e. g. "on", "LED", "motor", "blink") displayed on the blocks directly. The tags shown on the blocks were based on the tags that the children and young people saved in the database.

The prototype was a horizontal prototype, a partial working system. We used a mixture between cooperative inquiry as introduced by Druin [Druin, 1999] (that we discussed in more detail before in section 5.3 and section 6.1) and the *coaching method*. In the coaching method [Nielsen, 1997] an expert is allowed to help the test person to use the system properly. In our case, the interactor was allowed to help children and young people and give hints on how to use the software. The tests were usually performed with two persons: an interactor and a note taker. At least one of the persons was not directly involved in the development of the software. Additionally, we performed a log file analysis and asked subjective questions in an interview. From the answers to the questions we hoped to receive results to improve the recommender engine.

We also asked which view children and young people preferred in the questionnaires. The questions (in a translation from German) were:

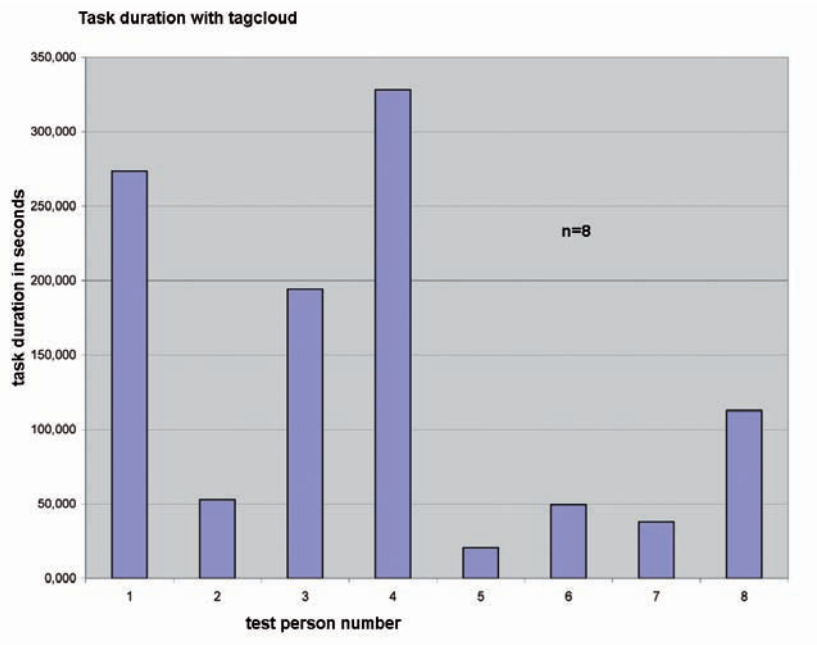
1. Did you work with Arduino or Handy Crickets or Lego Mindstorms before?
2. Do you use the Internet for chatting?
3. If yes, how do you recognize your friends there?
4. Did you prefer to see pictures of other children or their project or tags?
5. What do you find interesting in a project?
6. Why did you add a person as a friend?

7.1.1 Quantitative Data

From the log files we calculated how long the children and young people needed to perform each task and how often which element on the screen was clicked. The software was programmed with additional logging functionality to record these clicks on different elements and to log them. The first objective of collecting the data was to see if the children and young people needed longer to perform a task without tag-based programming. The logs were also used to reconstruct the events captured through the observation sheets. When comparing the young users who had a tagcloud displayed with further tags related to the single blocks to those who saw only the simple blocks labeled "On", "OnFor" and so forth, we had half of the users who completed the task (switching

on the LED) faster with the tags (see figure 7.1). The other half took even more time, so we do not have an indicator that the tagcloud helped the children and young people from the quantitative data (see also figure 7.2 to compare to users who did not see a tagcloud) . The data does not show a very clear picture, some children needed a long time to get used to the software till they finally clicked the “Upload” button (which we measured as task completion).

Figure 7.1: Task-completion time for the task “switch on an LED” in a view with the tagcloud.



The results from asking the additional questions were more interesting. We grouped the open answers into categories. Most children and young people preferred a project-centric view, had experiences with chatting and recognized their friends in the chat by their (nick-) names. In our software environment, they mostly added other users as friends, because their projects were interesting, advanced in their functions or aesthetically pleasing. Projects that used sensors instead of only actuators (e. g. T-shirts that blink in a certain pattern are not using sensors, while a T-shirt that blinks when the wearer makes a movement has sensors), were generally ranked as “more interesting”. The figures 7.4 and 7.5 show the results of the open questions in more details.

7.1.2 Qualitative Data

The qualitative evaluation of the usability testing used the data we could extract from the observation sheets. The children and young people used the system in different ways. Most of them read information about the projects and even wished for more information (when asked to specify, the young users brought the following examples: list of parts that were used, construction plans and the other user’s age and hobbies). An example is the following quote of a young girl.

Figure 7.2: Task-completion time for the task “switch an LED on” in a view without the tagcloud.

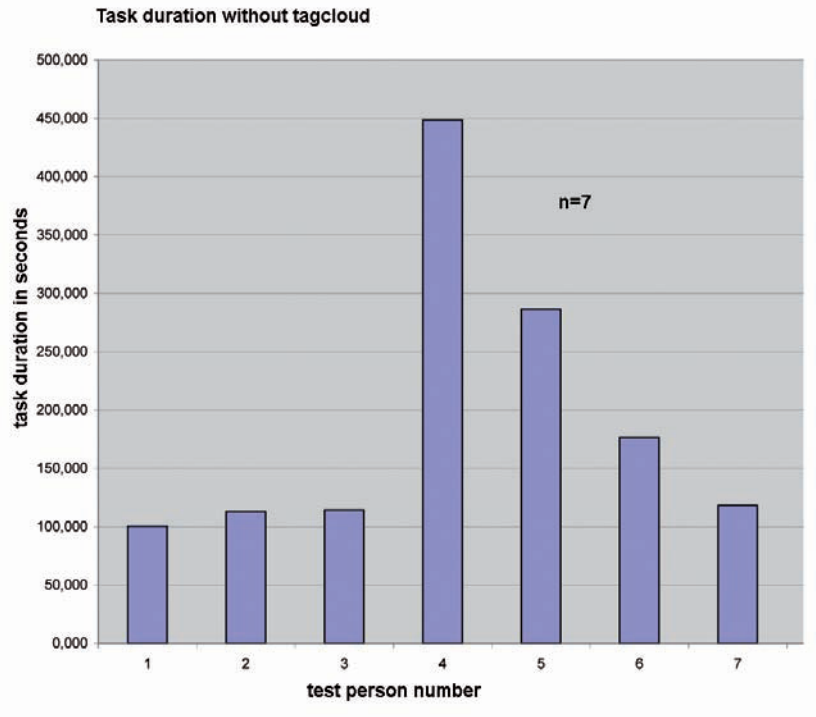
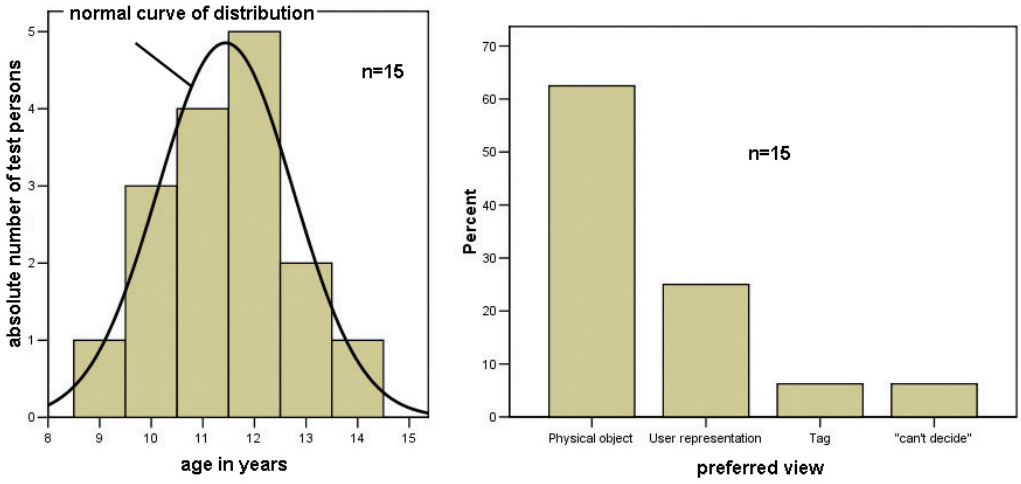


Figure 7.3: Age curve of the participants and their preferred view. Most children and young people preferred seeing the physical object.



Man weiss ja nicht, wie alt die sind. Und mehr Daten wären besser... Warum steht da nicht mehr Text?
[schoolgirl while working on task 2]

Figure 7.4: Diagram depicting the users' experiences in programming. More participants had prior experience in programming.

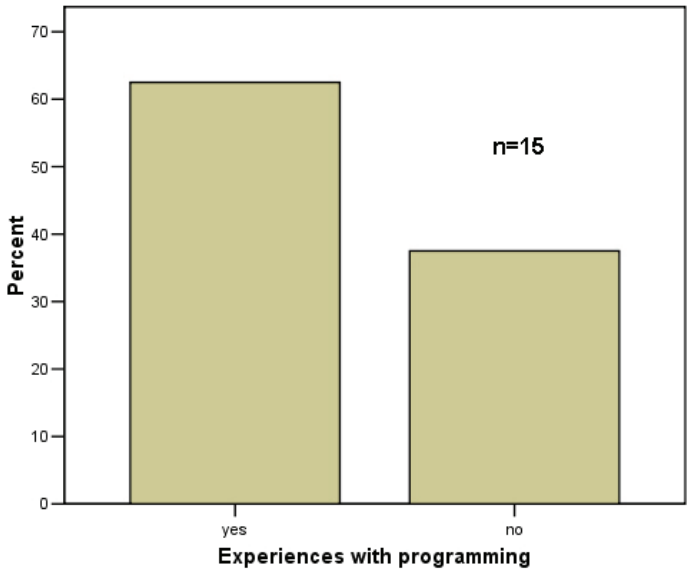
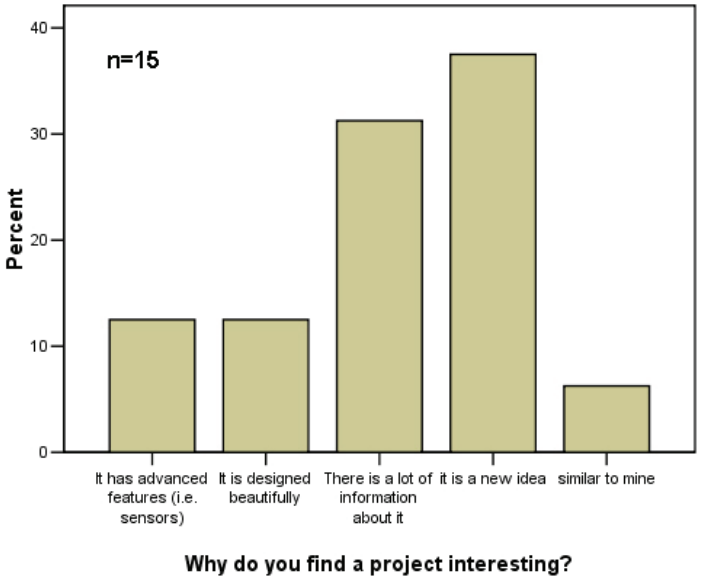


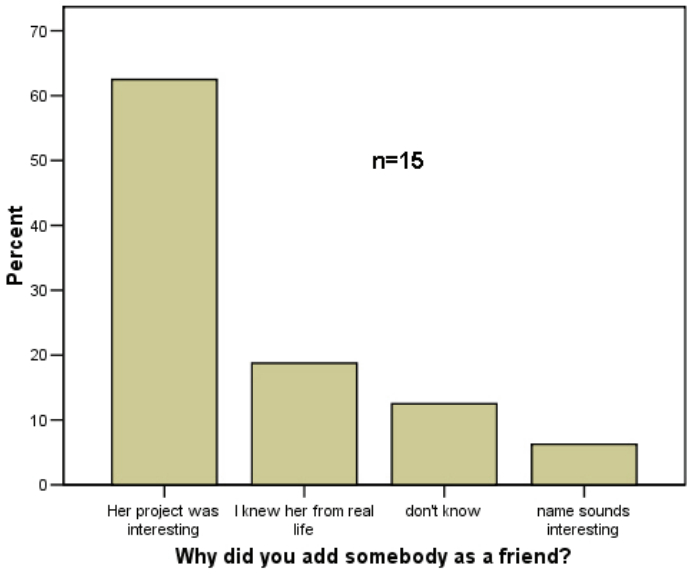
Figure 7.5: Diagram depicting the users' reasons for choosing projects in the tests.



> You do not know how old they are. And more data would be better... Why isn't there more text?
> [schoolgirl while working on task 2—translation]

However, some children and young people mainly relied on the visual information and would prefer video and audio information as well. The tags were used along with the visual information, possibly because they consist of very short text and are easy to read.

Figure 7.6: Diagram depicting the users' reasons for choosing friends in the tests. For most test persons, the project of the friend was most important.



> Ich gucke mir eher die Bilder an. Ich lese immer nur das [zeigt auf die Tags unter dem Bild]. Bilder sollten grösser sein, man erkennt kaum was... Beschreibung lese ich nicht. Video wäre besser.
> [schoolboy while working on task 1]

> I rather look at the pictures. I just read that [points to the tags displayed under the image]. The images should be larger, you hardly see anything... I do not read the description. A video would be better.
> [schoolboy while working on task 1—translation]

A lot of participants reacted strongly to images of projects with exclamations, e. g. "cool!" or "wow!". After seeing the pictures, they got distracted from their tasks and kept on browsing in the projects. They tried to understand what the project was actually doing by using the tags, pictures and descriptions. With a growing number of entries in the database, the users found the visualization overwhelming and wanted to see rather less projects, users or tags. Some suggested showing only similar projects to use as a model or to ask somebody working on a similar project for help.

In the group that was tested against tag-based programming, some participants read and used the tag information ("I knew which block to take, because it says LED." [schoolgirl while programming]), but sometimes also wondered because of the inconsistent information (e. g. "Is "Delay" a break or idling of the engine?" [schoolboy in an interview]). Some did not know words on the blocks (e. g. "actuator") and used tags instead gladly.

> Ich würde das nehmen [klickt auf den "Waituntil" Block] wegen 'Schalter', beim Anderen [refers to the "If" block] sagt er 'Werte abfragen'.
> [schoolboy while working on task 3]

> I would use that one [clicks on the "Waituntil" block] because of 'Switch', the other one [refers to the "If" block] says 'get values'.
> [schoolboy while working on task 3—translation]

> Nur "An" wäre da nicht so gut.
> [schoolboy using the "On" block tagged with "On", "Motor", "LED"]

> Only "On" would not be that good.
> [schoolboy using the "On" block tagged with "On", "Motor", "LED"—translation]

These examples show that the users did consider the tags when deciding which block they wanted to use. The boy in the first example thought about what tag represented something closer to what he wanted to do (use a switch). Some children and young people also considered the tags as helpful; in the second quote the boy clearly prefers seeing tags instead of only one label (here: "On").

7.2 Learning Experiences

To get an insight into what children and young people experienced in the different workshop scenarios, we will look at two participants in each learning scenario in a close-up look. The goal of these close-up looks is to see how the individual personalities with their background and experiences, and also with their expectations act in the workshops. How do they:

1. perceive material and concept in regard to the criteria flexibility, familiarity and appropriations?

2. relate materials and experiences to their everyday life and their future perspectives?

3. learn in terms of do they broaden their field of possibilities through the workshop?

Our hypothesis is that—as we suggested in our vignette—the Lego Mindstorms material might be perceived as inflexible and does not allow to connect to the life-world of popular culture and that our system is perceived differently in that respect.

For a more general comparison, we transcribed group interviews, contextual interviews and working situations where they seemed interesting for our questions. We used not only the material of the four participants, but of all children and young people that took part in both workshops. The transcribed data was coded. We identified different concepts and ordered them in the categories that we already discussed in the first four chapters of this thesis. We compared what the children and young people said in the different categories depending on the workshop they were referring to.

7.2.1 Technology Comes Alive

The first learning environment is quite similar to the one we summarized in our vignette. We looked into a weekend workshop in the youth hostel in Bremen. The participants worked with Lego Mindstorms and were asked to build robots with a basic model from the manual. We took this example for comparison, because with robotics it is in a domain we assume not connected to the children and young people's life-world, because the concept as well as the material are rather inflexible (construct with the help of the manual, use Lego Mindstorms) and the programming software does not take context or sociability into account. We compare this scenario to the one we developed with our design principles in mind.

All participants (twenty-one altogether) were boys aged between ten and fourteen (see figure 7.7 for details). It was not intended to organize a boys-only workshop, but only boys applied to take part. The topic of the workshop was *creatures* and it took part during three days (over the weekend), from 11 to 13 November 2005. The announcement focused on robotics, but rather on supporting the children and young people's own ideas.

> Das Herzstück der selbstgebauten 'Wunderwerke' ist ein programmierbarer Microcomputer, der die Motoren der Roboter über Tast- und Lichtsensoren steuert. Es werden aber auch Holz, Pappe, Stoff und andere Materialien verwendet. Der Phantasie sind keine Grenzen gesetzt, eigene Ideen können gleich ausprobiert werden und das Bauen und Programmieren am Computer ist kinderleicht.
> [Aus der Ankündigung des Robotik-Workshops]

> The heart of the self-made 'marvels' is a programmable micro-computer that controls the robot's motors via touch and light sensors. We will also use wood, card-board, fabrics and other materials. There is no limit for your imagination, your own ideas can be tested; building and programming on the computer is very easy.
> [from the announcement for the robotics workshop—translation]

The participants' experiences with technology were mixed. In an initial questionnaire we asked some questions about how they relate to technology in their real life and more than half of the children and young people had some experiences (see figure 7.8).

We already mentioned DiMeB's general workshop concept in section 6.1.1. As we pointed out, each workshop is a bit different but all of them implement the different stages in some way. This is why we roughly sketch the schedule of the workshops we analyzed.

We started the workshop with an introduction round and afterwards collected ideas and associations children and young people had connected to robots. The collection took part in a roundtable discussion; we collected the ideas—written on file cards—on a pin board. Afterwards, we allowed the participants to experiment with three different prebuilt robots and discussed how they worked. It was the children and young people's task

Figure 7.7: Age of the participants

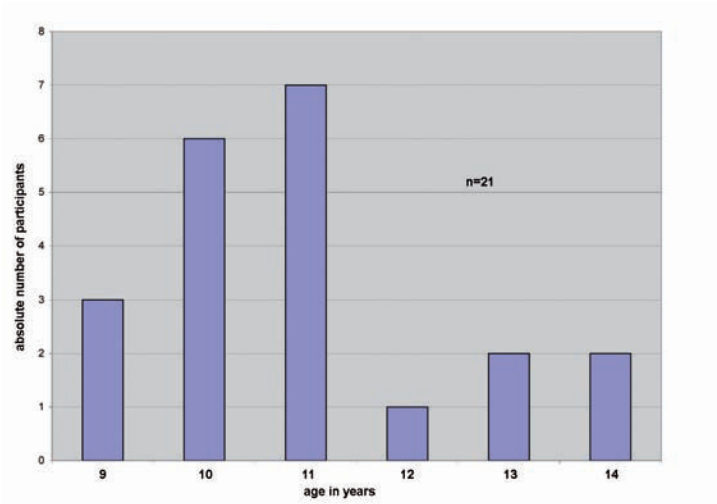
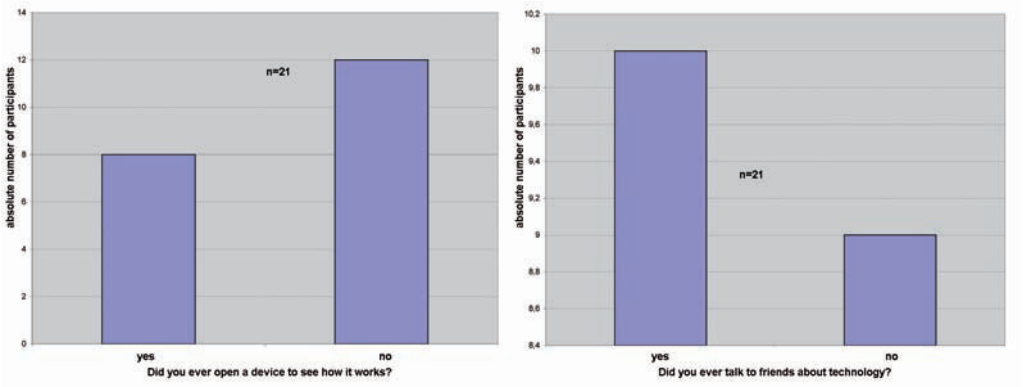


Figure 7.8: On the left: Did you ever open a device (e. g. a radio) to see how it works? On the right: Did you ever talk about technology with your friends?



to find this out. In this workshop, the activity implemented the *introduction to hard- and software and material* stage of the general concept.

The participants then formed subgroups and worked on their own ideas, sketched the ideas and started building them. When most groups were finished with building the robots, we introduced programming with a game. The children and young people could try to "program" their tutor who understood only certain correct commands. Afterwards, the participants "programmed themselves" and eventually programmed their robots. From time to time, all participants and tutors gathered to present their current projects and developments and to discuss how to proceed further. The workshop ended with a presentation where many parents and friends came to see the outcome. The participants' artifacts were presented in a kind of a show where each group described and showed their project on stage.

Four tutors (excluding the author) supervised the workshop, three were male and one was female. Two of the tutors were international students that hardly spoke any

German. Questionnaires (cf. appendix B) were handed out at the beginning of the workshop. Group interviews were conducted after the workshop; two subgroups were interviewed together in each interview session.

The author observed the workshop, took notes, videotaped and conducted group interviews (that were semi-structured). The children and young people perceived the author as part of the workshop team and came to her with technical questions or with other topics they needed support with. Therefore the interviews were rather informal.

7.2.2 Case Study: "Lego Is Stuff for Children, for Babies"

We start with a closer look on Andy—whose name we changed—and his experiences in the robotics workshop. Twelve-year-old Andy was a boy who tried to make his point very clear and articulated his ideas as precisely as he could. In group discussions he stayed silent though, but when asked individually it was obvious that he followed discussions and generally what was happening in the workshop. Andy came to the workshop, because he read information about the workshop in the newspaper himself, opposed to other participants whose parents suggested the workshop to the youngsters. He remembered something like:

> Man gibt ihrem Kind die eigene Imagination oder so.
>
> Your child receives her own imagination or something like that.
> [translation]

The sentence caught his eye and he looked forward to building his own robot. Andy was interested in technology, but when he moved to Bremen he did not find opportunities where he could learn more about the topic.

> Also, das [etwas mit Technologie] war eigentlich mein Traumberuf, aber ich dachte das wird nie klappen und wo soll ich das auch lernen. So was gibt es ja überhaupt nicht in Bremen. Dann wollte ich das nicht mehr. Jetzt will ich einfach Volvo Chef werden. Aber jetzt habe ich das in der Zeitung gefunden, jetzt ist das Interesse wieder etwas gestiegen.
>
> Well, this [something with technology] was my dream job, but I thought that will never happen and there was no opportunity to learn something like this in Bremen. Then I did not want it anymore. Now I want to become Volvo CEO. But now, after I found this in the newspaper the interest was raised again.
> [translation]

When the workshop started Andy was slightly disappointed, because they were supposed to work with Lego. For him

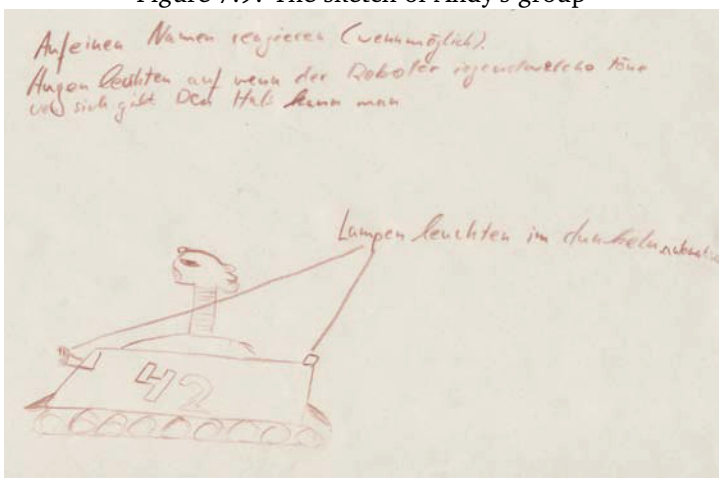
> Lego ist so ein Kinderkram, so Babykram.
>
> Lego is stuff for children, for babies.
> [translation]

What he imagined beforehand, was building something out of metal parts. When he came to the workshop, he had no clear idea what a robot was, but he knew many examples: combat robots for the field, robots for play, robots that work in nuclear power stations or in medical applications. There was also the Sony Aibo[1] and the Furby[2] Andy categorized as robots. For Andy, the common thing about these different robots was that all robots could work where humans cannot. When the subgroups formed Andy was also astonished, because he imagined everyone would build their own robot.

> Jeder hat einen eigenen Roboter usw. das habe ich mir vorher so vorgestellt. Also, in der Zeitung stand ja nicht, dass man Gruppenarbeit macht.
>
> Everyone has her own robot and so on. That was what I imagined. The newspaper did not say we would work in groups.
> [translation]

When the subgroup started to work he teamed up with Lan, the international tutor from China, who hardly spoke any German. Andy was fascinated by this fact and started to talk to Lan trying to improve his English skills. He could work better with the tutor than with the two other boys in his group. When the group started sketching their ideas, they came up with something very un-related to creatures, namely a tank robot that was supposed to react to a name, move its head, play melodies and blink with its eyes (see figure 7.9 for illustration). Andy stayed rather passive during the building process; he

Figure 7.9: The sketch of Andy's group



happily searched for parts that Lan required. In the programming game he did not take part actively, but he still watched it intently.

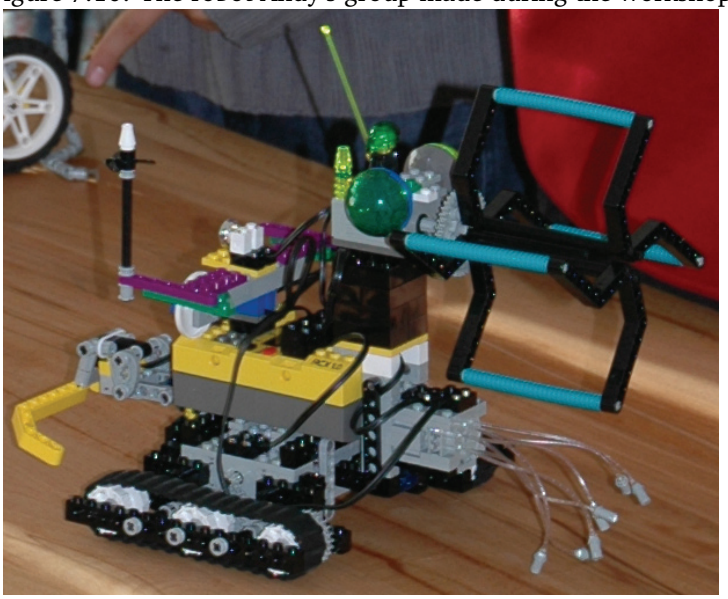
Andy tried to work with the boys from his group, but he was often ignored. He tried to get hold of the robot, but often the other boys dominated and did not let go of the robot. We can see that very clearly on the video tape. In the interview Andy always insisted that there were no conflicts in the group.

[1] http://support.sony-europe.com/aibo/
[2] http://www.mimitchi.com/html/furby.htm

> ... wenn man Gruppenarbeit macht, wenn jetzt einer dafür ist und der eine dagegen, dann kommt es manchmal zu Problemen und man will nicht zusammenarbeiten. Das ist bei uns nicht vorgekommen.
>
> ... if you work in a group, if someone is in favor and someone against, you sometimes have problems and you do not want to work together. That did not happen in our group.
> [translation]

The positive thing for Andy in his group work was that he was appointed as speaker of the group and presented the group's project when the children and young people met and also in the final presentation. The group made a crawler-mounted vehicle that had a light sensor and could turn its tail when the light was bright (you can see the final result in figure 7.10). When it came to programming, we observed that Andy had problems to

Figure 7.10: The robot Andy's group made during the workshop



grasp the concepts. Still, he felt he got programming right and figured out how it worked himself.

> Ich habe das selber rausgefunden, also vorher habe ich das nicht verstanden, aber dann habe ich das verstanden. Als ich es dann verstanden habe, war es auch ganz leicht.
>
> I discovered this myself, before I did not understand it but then I did. As soon as I understood it was very easy.
> [translation]

Although he had this impression, he could not make the connection between the programming he did with Lego Mindstorms and the robots he knew already before the workshop as you can see in the following quote:

> Ja, Aibos, die können ja auch selber denken. Sie haben auch mehrere Sensoren. Also, sie sind fast wie ein echter Hund. Also, sie können auch selber denken, also ist das gar nicht so schlimm, oder?
>
> Well, the Aibos can think themselves. They also have many sensors. But they are nearly like a real dog. They think by themselves. That is not that bad, is it?
> [translation]

Programming seemed hard for Andy (a contradiction to his statement we quoted above) and he wished the robots could understand him better (the language to program the robots would be easier). He wished

> Dass er Dinge nicht so schwer versteht, sondern, dass er sie sehr einfach versteht.
>
> That understanding things for him was not that difficult.
> [translation]

When we look closer at Andy's statements we can see that he related Lego to the world of play, while the real world of work with metal parts was imagined as different and working there was really difficult compared to working with Lego—in Andy's impression.

> Interviewer: Jetzt wo du das [den Workshop] gemacht hast, kannst du dir auch eher vorstellen das [einen Beruf im Bereich Technologie] zu machen?
>
> Andy: Also ich kann mir nicht so richtig vorstellen, dass man da so richtig Geld kriegt, wenn man das arbeiten will. Solche Roboter bauen oder so etwas in der Art.
>
> Interviewer: Ja also, man kann ja z. B. grosse Roboter in einer Firma
>
> Mehmet [unterbricht]: verkaufen?
>
> Interviewer: bauen
>
> Andy: Ja, aber das ist dann bei so was grossem ganz also äh... [Einwurf: kompliziert?] ja, das ist sehr kompliziert mit so Teilen und grosse Maschinen zu kriegen, da ist das doch wohl eher angebrachter wenn man so richtigen Strom hat, so mit Stromkabeln und Metall und so arbeitet. Aber bei so was Kleinem ist es doch besser wenn man Lego hat.
>
> Interviewer: Now that you took part [in the workshop] can you imagine doing something like that [a technology related profession]?
>
> Andy: Well, I can't imagine that you earn real money when doing something like this. Building robots or something.
>
> Interviewer: Well, but there are big robots in a company you could
>
> Mehmet [interrupts]: sell?
>
> Interviewer: construct.

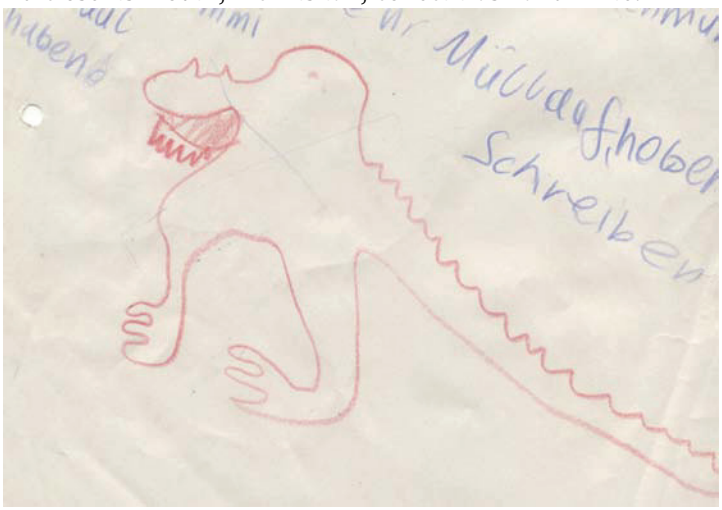
> Andy: Well, but with big things it is so, erm... [objection: complicated], yes, complicated with parts and big machines. It is better then to take real power and cables and metal to work with it. For something small, Lego is fine. [translation]

Even when looking only at the few quotes we discussed here, we can see how the different materials raise different associations. Programming and building Lego robots is nothing Andy connects to the world of work. You do not earn money with something like this. The big robots are *real* and you need real materials (Andy thought of metal as such a real material) to build them, which is more complicated. Andy connects to the world of everyday life, but only to the world of play and cannot transfer the new knowledge to wider parts of the world around him. Programming—opposed to constructing the robots—is perceived as hard, the robot does not understand an easy language.

7.2.3 Case Study: "You Wanna Build a Dinosaur and out Comes a Tank"

Mehmet (we also use a different name here)—an eleven-year-old boy—came to the workshop, because his cousin was one of our tutors. He visited his cousin at another occasion (also a robotics workshop with Lego Mindstorms) and the workshops looked interesting to him. Mehmet was from an immigrant background; he grew up in Germany and spoke German though. Mehmet came with a friend to the workshop. When collecting ideas what a robot was, Mehmet came up with things that can move, help with cleaning up one's room or help in the kitchen. When the participants started experimenting with the ready-built Lego robots, he wanted to change from the creature's topic to build a field robot and asked for field robots again and again, but then, together with his friend, sketched a dinosaur the boys wanted to work on (see figure 7.11). The task was

Figure 7.11: Mehmet and his group's dinosaur sketch. The dinosaur was supposed to move, open and close its mouth, flick its tail, collect trash and write.



difficult to fulfill with using basic models from the manual, the boys ended up with using

a model that could grab things. The friends worked on the robot collaboratively, what was important for Mehmet. The tasks were very challenging for him, so he needed his friend's help.

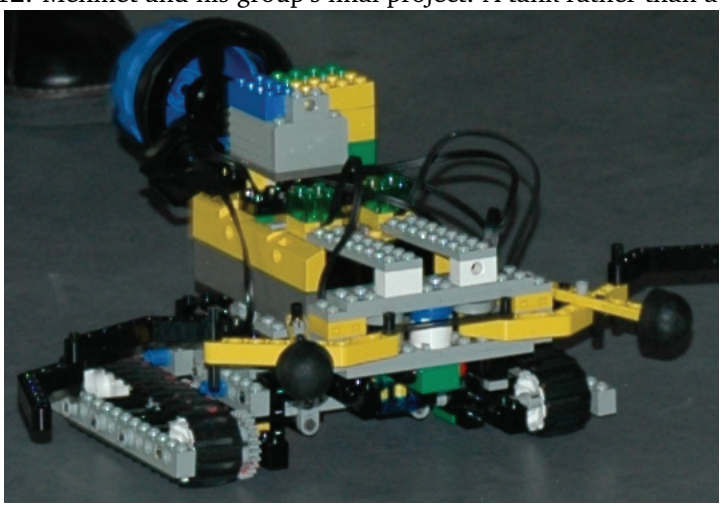
> Ja, einzeln wäre das dann zu schwierig, hätte keinen Spass gemacht, weil z.B. du sagst zu deinem Freund, ich habe zu F. gesagt, das kommt dahin und ham wir dann so gesprochen und ham geredet. Wie das aussieht und so. Das macht schon mehr Spass als alleine.
>
> Well, it would have been too difficult for a single person, would have been no fun. Because, you say to your friend, I said to F. that belongs there and then we talked. How it looks and so on. That was more fun than on your own.
> [translation]

Even building the robot with the help of the manual was a challenging process for Mehmet. The boys did not succeed in building what they imagined, their dinosaur. Mehmet wondered about that.

> Das willst du so machen, du willst zum Beispiel einen Dinosaurier machen und raus kommt ein Panzer. Also das war ganz komisch.
>
> You wanna do something, for example you wanna build a dinosaur and out comes a tank. That was strange.
> [translation]

In their case that was a good observation as you can see when looking at their final outcome in figure 7.12 that does not have a similarity to a dinosaur. When it came to

Figure 7.12: Mehmet and his group's final project. A tank rather than a dinosaur.



programming the activity was even harder for Mehmet than building the robot. At first, Mehmet was fascinated that it is possible to program Lego Mindstorms. He did not know that something like this was possible before.

> Ich habe auch gestaunt wie die fahren. Die fahren richtig. Ohne das man so mit Joysticks lenkt. Fand ich auch mal interessant.
>
> I really wondered how they moved. They really moved. Without being controlled with a joystick or so. Found that interesting.
> [translation]

During the game phase when the children tried "programming" their tutor, Mehmet took part actively and managed to give easy commands like "on", "stop", "left", "stop", but did not go into commands with "until" or "if". When they were supposed to transfer what they learned during the game, Mehmet had problems. Mehmet and his friend needed a tutor around all the time to help them programming. Even then, they seemed only to understand that a certain alignment of blocks had an effect on the robot, but not why and how both were connected.

> Also bei unserem Roboter, da wollten wir ein ganzes Kreis machen. Hat auch geklappt. Erst musste er geradeaus, dann musste er ein auf acht, ein auf eins, dann würde er sich so drehen, dann müssen wir ihn wieder geradestellen, acht, acht, dann ist der wieder und dann müssen wir wieder ein acht. So haben wir einen Kreis gemacht.
>
> Well, we wanted to make a whole circle with our robot. Worked out. First he had to go straight, then on with eight, on with one, then we had to make him straight again, eight, eight, then he is straight, then we have to do on with eight. This is how we made a circle.
> [translation]

In the quote you can see that Mehmet refers only to the programming environment ("on with eight"...), not to the robot or the motor they actually control (like "we switched our motor on"). While other children (see Andy's close up for example) could make a connection between Lego and "real" robots or jobs related to technology (even when they thought Lego is for play only), Mehmet did not connect the different worlds. In the (group) interview he always related to Lego, whatever the other participants said.

We will further analyze the examples in section 7.2.7 in relation to the second scenario that we studied.

7.2.4 Smart Fashion

Our second learning setting was a workshop organized within the EduWear project, and it was called *smart fashion*. The workshop took place at the DiMeB research lab, stretched over five days, during the holidays children and young people had around Pentecost 2008 in Bremen. The workshop was announced as smart fashion workshop, one of the tutors was a fashion designer in training. The announcement targeted children and young people with an interest in design and fashion and tried to take away initial "fear" of technology.

> Das Verhalten deiner Kleidung steuerst du mit einem Mikrocontroller, der gerade einmal so gross ist wie eine Brosche. Doch zuvor muss dieser programmiert werden. Aber keine Angst – wir benutzen eine spezielle Programmierumgebung für junge Menschen.
> [aus der Ankündigung des Smart-Fashion-Workshops in einer lokalen Zeitung]

> You control your clothes' behavior with a micro-controller that is brooch sized. But it has to be programmed before. No need to be afraid—we use a special programming environment for young people.
> [from the announcement of the smart fashion workshop in a local newspaper—translation]

The smart fashion workshop also implemented the different workshop stages. It began with an introductory round where participants and tutors introduced themselves and briefly explained what they expected from the workshop. Then the topic *intelligent clothes* was introduced and ideas what intelligent clothes could be, were collected with file cards at the pin board. The participants were divided into small groups and they visited different *stations* where they got to know sensors, actuators, micro-controller and different textile materials. The children and young people then progressed to work on their own textile circuits by embroidering simple circuits. From that, first ideas what kind of project they wanted to work on evolved and the participants started to sketch their artifacts. They started working on the artifacts; later programming was introduced with a game similar to the one we mentioned before. Each player represented a block and passed on a signal. Afterwards, the children and young people kept on working on their artifacts, occasionally starting to program. Each day ended with a roundtable discussion where every group presented their results. A deeper group discussion took place on the last day before the final presentation. The workshop ended with a presentation, the participants insisted that it should be a cat-walk style fashion show with a fair-like presentation afterwards.

Questionnaires (cf. appendix B) were handed out at the beginning and the end of the workshops. In addition to the group discussion, contextual interviews were conducted while the children and young people were working on their projects. The author did not attend the workshop but relied on video-taped contextual interviews, group interviews, questionnaires, video-taped working situations and reports from the tutors and the colleagues who observed the workshop. We chose two cases to study further, because we could find clear evidences that programming was important for the girls and because the girls' projects were very different in their technical complexity.

7.2.5 Case Study: "It Looks Really Cool"

The girl we will refer to as Anne read about the smart fashion workshop and was determined to go, because she was really interested. Since she went to school in another province she even had to get a special permission, because she missed some days at

Figure 7.13: Age and gender of the participants

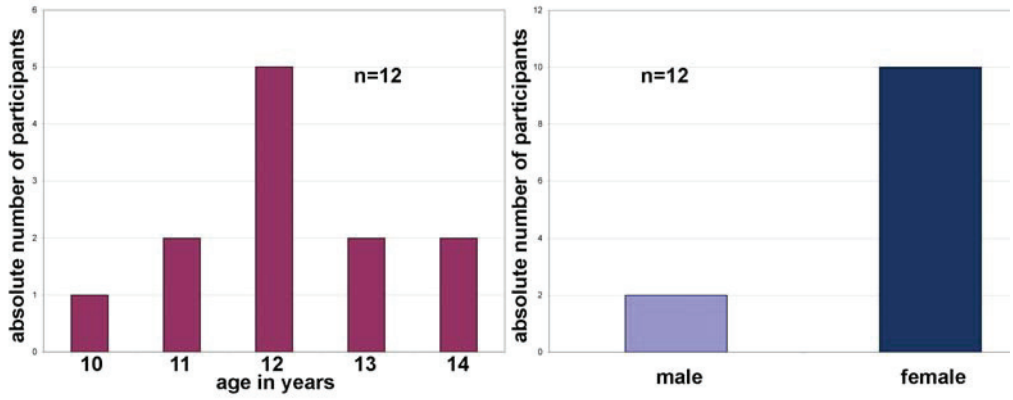
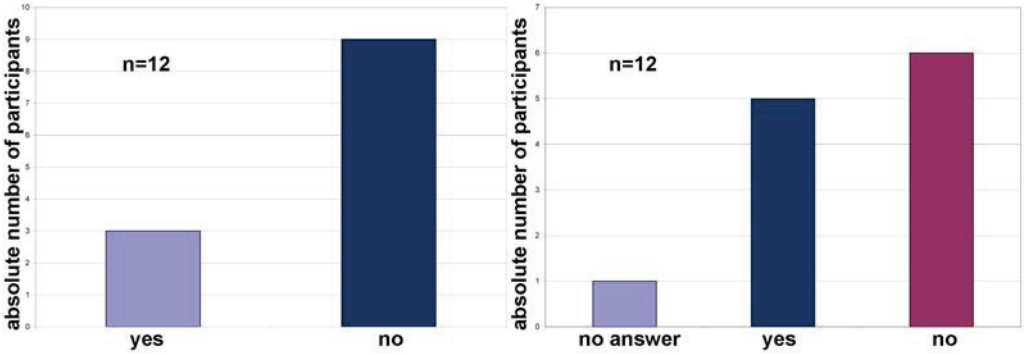


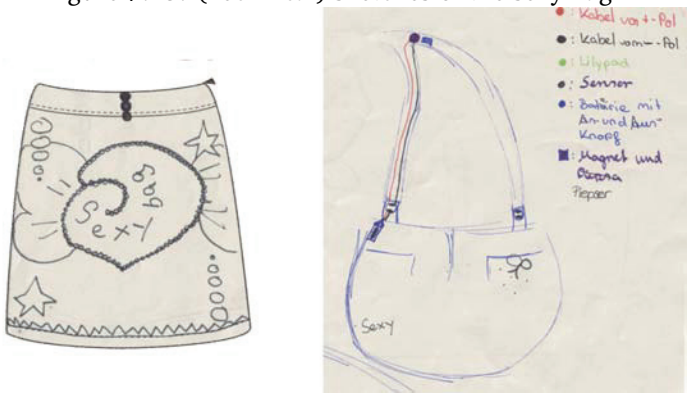
Figure 7.14: On the left: Did you ever open a device (e. g. a radio) to see how it works? On the right: Did you ever talk about technology with your friends?



school. Anne came with a friend to the workshop, they were among the older participants of the workshop—Anne was fourteen. Her initial idea was to make a garment that changed music according to the wearer's mood. The two girls sat down with two other girls when doing the first exercises and teamed up with them. Together they developed an idea for a bag that had an alarm attached. When someone stole the bag an alarm that consisted of a loud beeper and a red LED should be triggered. The girls combined the bag with a shirt. The shirt had a patch of conductive fabric sewn to its shoulder and closed a circuit on the holder of the bag. The whole circuit with controller, LED, beeper, switch and battery was inside the bag (see figure 7.15 for a rough drawing of the circuit).

> Und ursprünglich sollte das LilyPad auch hier sein, im Nacken. Das sitzt jetzt also auch an der Tasche, weil die ganze Elektronik wollten wir ursprünglich auch in den Bügel machen, aber der ist halt ran genäht und die kann man auch super in den Taschen verstauen und darum ham wir das alles auf einmal gemacht.
>
> Originally the LilyPad was supposed to be here, at the neck. But now it is also in the bag, because the electronics were supposed to go into the hold, but it

Figure 7.15: (Technical) sketches of the sexy bag



> is sewed on and the electronics fit into the bag very well, so we did it all at once.

Anne really wanted to make cool pieces of clothing and it was a great group for her, because Lisa—one of the newfound friends—was an expert sewer with her own sewing machine at home. Together they designed further and further in a very creative process. Anne brought old jeans to the workshop that she quite liked and the jeans then served as basic material for their bag. Somebody in the group was always coming up with a new idea how to decorate the bag or shirt. Whenever Anne was waiting for somebody else in the group to finish something, she kept on decorating the bag.

> Und eigentlich auf der Tasche hatten wir so aus Spass mit Pailletten so “sexy bag” drauf geschrieben, das ist gerade so ein bisschen abgegangen und dann haben wir das gestern noch hier [zeigt Shirt] das noch mit Stofffarbe drauf geschrieben und dann noch ein Bügelbild gemacht und so.

> On the bag we wrote “sexy bag” with sequins, just for fun. This came off a little and then we wrote this [shows the shirt] with textile paint and applied an iron-on patch.
> [translation]

As an assignment for school Anne had to make a video so she decided, she could use the opportunity and make a video that demonstrated the features of their bag and T-shirt. They used the last day to make the video at the university. Programming proved as a rather easy task for Anne, she had imagined it much more difficult. For her, there were new possibilities in programming because she—all of a sudden—could solve problems that were impossible to solve with circuitry only.

> Dann wäre ja z.B. wenn der Stromkreis geschlossen wäre, dann würde ja die Alarmanlage theoretisch normalerweise angehen, weil die dann ja Stromzufuhr hat. Und um dem halt entgegen zu wirken, da müssen wir halt programmieren und den Sachen sagen, dass sie was anderes machen sollen.

Figure 7.16: The bag as a part of Anne's project. Most electronic parts including the micro-controller were hidden inside the bag.

> When the circuit was closed, usually the alarm would be switched on because it would be powered. To counteract, we have to program and tell the things that they have to do something else.
> [translation]

Anne was very interested in design and aesthetics and the material we used in the workshop gave her the opportunity to focus on these aspects. On the other hand, their project was technically advanced and Anne understood a lot of circuitry and programming, both topics and activities she had very little previous experiences with. Anne really connected the material to her life-world aiming at producing something she would wear in reality.

7.2.6 Case Study: "Wow, I Can Make Them Blink"

Twelve-year-old Janina (once again we changed the name) came to the workshop with a clear idea in mind: She wanted to make a shirt with stars on it. The stars were supposed to lighten up as you can see in figure 7.17, an early sketch of the project. Before the workshop Janina thought they would possibly connect some lights to a battery that would then light up when being switched on. At first, she was a bit disappointed that they had

Figure 7.17: A sketch of the star shirt. The main idea behind the project was to create a shirt with stars that could light up whenever the wearer wanted them to.

to use the LilyPad, but that they were not allowed to take it home. Then she was quite enthusiastic, because of the new possibilities the small controller offered. Janina found

Figure 7.18: Working on the star shirt. A lot of sewing was required in the project.

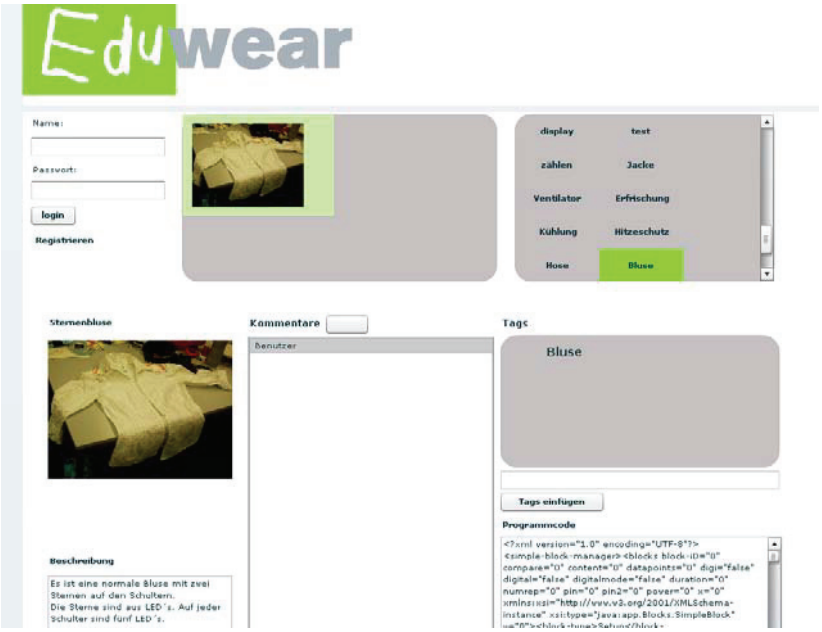
out that she had to sew a lot, especially because she made mistakes and had to start again from the beginning a couple of times.

> ... weil wenn man nur eine Sache falsch hat musste man alles aufmachen, bei dem einen Stern hätte mir nur so ein Stück gefehlt, ich war ungefähr schon hier, dann habe ich das aber so zusammengenäht, dann musste ich alles wieder aufmachen. Und das Ganze zweimal.

> ... because even if there is only one thing wrong you have to untie everything. There was only such a piece missing with one of the stars, I was already here, but I sewed it together so I had to untie it. And that happened twice. [translation]

Still, it was a great experience for Janina. She never thought she could make LEDs blink with some programming that was also quite easy. Another thing she had never thought of was the very short time in which she could achieve such a result.

Figure 7.19: The star shirt project in Amici's browser-based client. The browser-based client allows the public to see different projects (by choosing tags from the tagcloud on the right) and comment on them (the box in the middle frame).

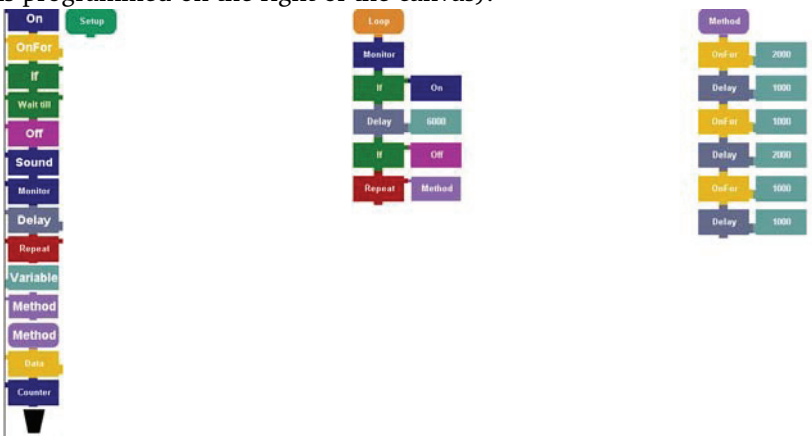


> Also ich hätte nicht gedacht, dass man in zwei Stunden gleich Lampen einbauen kann. Und die dann auch mit dem LilyPad aufblinken lassen kann. Ich dachte, dass man dann die Lampen nur so leuchten lassen kann, wenn man gerade will, aber nicht, dass man die Blinken lassen kann.
>
> Well, I did not think that you can build in lights within two hours. And that you can make them blink with the LilyPad. I thought you can only switch the lights on, but not that you can make them blink.
> [translation]

Janina could remember everything that happened during the programming game quite accurately and related it to her own program (see figure 7.20). Only the method block was a challenge.

> wir haben ja dieses eine Spiel gespielt übers Programmieren und da hat sie [die Tutorin] gesagt, dass ich da ganz gut aufgepasst habe. Und bei uns kamen auch zwei Sachen dazu, nämlich die Methode, da weiss ich immer noch nicht so richtig was das ist, aber das war nur wichtig, dass eine mit dabei ist.
>
> We also played that game and she [the tutor] said that I paid attention. And then there were two new things in our project namely the method. There I

Figure 7.20: The star shirt's program. In the main loop (in the middle of the canvas) sensors are asked for their values (with the if block) and depending on the value LEDs are switched on or off. In each loop a method is called that starts a blink pattern (the method is programmed on the right of the canvas).



still do not really know what that is, it is only important that there was one. [translation]

The programming was perceived as quite easy; still Janina felt dependent on the help of a tutor.

> ... ist es eigentlich ganz einfach. Aber wenn man eben manchmal überlegen muss, aber ich hatte ja zum Glück Eva dabei, da konnte ich immer das, das, das sagen und ich hatte auch fast alles richtig.

> ... it is actually quite easy. But sometimes you have to think, I was lucky that Eva was around, then I could say everything and I was usually right. [translation]

Janina could well imagine wearing her shirt in her real life.

> Also, ich würde das auch ganz normal zur Schule tragen.

> Well, I would wear it at school. [translation]

Janina was very enthusiastic about the new technology and the new possibilities she had through what she learned during the workshop. Her field of possibilities definitely widened through the new experiences.

7.2.7 Categorization of the Outcomes

As we mentioned before, we looked at the four childrens' experiences in more detail but also analyzed data of all participants in the workshops. We went through the transcripts and coded them with open coding [Bryant and Charmaz, 2008]. Then we ordered these

open codes into the categories we discussed in our concept: flexibility & constraints, familiarity & associations and appropriations & contextualizing. We also looked at how the children and young people perceived their learning process. Table 7.1 gives an overview of categories, concepts and quotes. The quotes are only examples to illustrate what the children and young people said and how we categorized it. To make reading easier, we used translations from German as quotes. We mentioned our hypotheses already in section 7.2. They were:

1. The concept and the materials in the robotic workshop feel restricting for the participants.

2. The EduWear construction kit relates to the participants' life-worlds and to popular culture.

3. Amici makes programming fun and a social activity.

The outcomes of this qualitative case study cannot be generalized and we can't (and do not want to) compare the different settings and materials coming to a final conclusion which one is "better". The interesting results are the reasons how and why the individuals perceived the material and activities in the way they did.

When we look at main differences between the two workshops, we can see that:

1. Andy and Mehmet found programming was hard (references to how hard it was to make the robot understand), while Anne and Janina in smart fashion were rather surprised how easy programming was (direct statements like "it is actually quite easy").

2. In the robotics workshop, the participants related the material to their world of everyday life, but rather to the world of play. The world of work was perceived as very different. Our cases in smart fashion related the material to their world of everyday life and to popular culture (they wanted to "pimp" cool things), for example most of them could imagine wearing their artifacts at school. They also made a connection to their experiences in the workshop and school or future profession (see the example quotes in table 7.1).

3. Lego Mindstorms (as well as the workshop's concept) were perceived as a material that limited the children and young people (they did not have the freedom to build what they wanted to). The textile material on the other hand, was not perceived as limiting, the ideas were changed, because of the difficult activity of sewing and because of the concept (mainly because the participants were not allowed to take the LilyPads home).

4. In smart fashion, appropriations or contextualizing of the material played an important role. The aesthetic dimension had an impact on whether or not the participants could imagine wearing their artifacts. Hiding the technology was extremely important to all interviewees. We could not find a reason for this in the material. Once it looked good and you could not see "cables", the children and young people

could imagine wearing their artifacts because they were proud of them, the intelligence made the artifacts something special or simply because of the function of the artifact. A reason why some participants would not wear the artifact was that they might lose control and the artifacts would beep around unintentionally or because the electronic parts, i. e. the battery, made the garment uncomfortable.

Phenomenon	Quotes
Flexibility and Constraints	
Limitations of the material	"Well, this did not work well, we would have to make another circuit on the garment and connect itto the LilyPad." [smart fashion]
	"We did not use it yet, because we still have difficulties with the contacts." [smart fashion]
	"That does not work with Lego". [robotics workshop]
	"I would have liked to work with metal. You have more freedom then." [robotics workshop]
Limitations of the concept	"We did not have real freedom, because we had to build what it said there." [robotics workshop]
Familiarity and Associations	
Previous experiences	"Did you program before? No. Never." [smart fashion]
	"But you never programmed before? At home or at school? Well, I think I did once at school but I am not sure." [robotics workshop]
	"I sewed at home and at school and I have my own sewing machine at home." [smart fashion]
Expectations	"No, I imagined it somehow differently. That we do not use Lego but metal parts and so on." [robotics workshop]
	"Well, at first I imagined it very easy, but I did not know that there are conductive fabrics and yarns. I thought you take cables or something. Well. Then I realized that it was much harder. Then I thought it was even harder than it was in reality. It is exactly right, you can do it if you do it properly." [smart fashion]
World of everyday life	"There are these buzzers for grandmas." [smart fashion]
	"Well I would wear it [the smart artifact] to go to school." [smart fashion]
Popular culture	"Made the T-shirt a bit funnier, we pimped it." [smart fashion]

	"Because I think it is quite cool, the way it looks." [smart fashion]
World or work	"Well, I also thought that you can use a lot. The programming, the sewing or generally how a sensor works and I think we will need this later in our jobs." [smart fashion]
	"I can't really imagine that you can earn real money doing something like that. Building robots or whatever." [robotics workshop]
World of play	"Lego is stuff for children, for babies." [robotics workshop]
Potential in the material	"Someone came with something, a piece of fabric or also with this string [shows string on the bag]. Then somebody made a bow and I just fixed it, looked at it and it looked good." [smart fashion]
Potential in the programming	"I really wondered how they moved. They really moved. Without being controlled with a joystick or so. Found this interesting." [robotics workshop]
	"I didn't think that you can build in lights within two hours. And that you can make them blink with the LilyPad. I thought you can only light up the lights when you want, but not that you can make them blink." [smart fashion]
Appropriations and Contextualizing	
Aesthetics	"We only have the conductive yarn here but if it was too much and that would not look nice I would not wear it at school." [smart fashion]
Hide technology	"That would have been too much cables on the outside. You could not wear it."
Personalize	"Then we can make the T-shirt something special." [smart fashion]
Make something special	"I would wear it, because it is something different." [smart fashion]
Comfort	"I would not really wear it, because I think with the batteries it is a bit too heavy and just not comfortable." [smart fashion]
Function	"And an alarm is useful, isn't it?" [smart fashion]
Learn and Design Process	

Field of possibilities	"I think we got a lot of new information. With the conductive yarn and fabrics, I did not know that there is something like this." [smart fashion]
	"In the beginning I did not believe that we can make a shirt intelligent in only four days. Or that one can build such a circuit. I though that was, wow." [smart fashion]
Programming	"I thought programming was much more difficult." [smart fashion]
	"I wish that understanding things for him was not that difficult. " [robotics workshop]
Future	"You can use this [the knowledge] later on. You can understand things better after you did them. And for later for the job and life, too." [smart fashion]

Table 7.1: Examples of quotes and their coding

7.3 Gender

We did not discuss the topic gender in more detail up to now. We mentioned that smart textiles were used as a medium to raise girls' interest in technology (cf. section 3.8.3). Although gender issues are not a focus of this work, we would like to allude some points we noticed. In our example workshops, we find a very conventional allocation: Only boys took part in the robotics workshop, while in smart fashion there were mainly girls. These are extreme examples (during the EduWear project we also had smart textiles workshops with boys only) but confirm what we notice quite often—the announcement of a workshop really matters and robotics do not seem to attract girls as well as boys. Smart textiles—when the workshop focuses on fashion—rather attract girls (cf. Buechley [Buechley, 2006] who reports similar experiences).

We also noticed that the aesthetical aspect was usually more important for girls than for boys. Nevertheless, this focus did not have an impact on the technical complexity of the projects the children and young people worked on.

7.4 The Tagging Process

To evaluate the tagging process, we will discuss some examples of projects and tags in our database (see section 7.4.2 for more information on the database's content) and bring them together with the observations we made when we supervised the tagging process or (as in the usability tests) the project retrieval. Some projects were published unsupervised, some within our workshops with support of the tutors. We also interviewed two of the more active taggers (a fourteen-year-old girl and a ten-year-old boy) in a guided

interview (see appendix B.2) while showing them some projects in the database at the same time. This way, the interviewees were reminded of their own projects and tagging process. We also interviewed a less advanced tagger, a thirteen-year-old boy who tagged for the first time when uploading his project.

Let's now discuss some projects children and young people tagged as a case study. For example, we can find a project in the database that is called "Schatz" (meaning "treasure" in German). The project was part of an adventure story children and young people told. When they told the narrative on stage they were supported by a number of physical artifacts that were part of the story. The "Schatz" opened up when the user entered the right combination to access the treasure chest. The project (the device to enter the code) is tagged "Rätsel" (German for "riddle" or "mystery"). When we take up taxonomies of tagging systems that we discussed in section 5.1.3 we can interpret the tag "Rätsel" as "identifying what it is" in Golder and Huberman's taxonomy. The user incentive based on Marlow et al.'s taxonomy was "future retrieval" according to the interview with the taggers.

Another example is a project called "LCD-Bildschirm" (German for "LCD-screen"). The project contained code to write letters to a special hardware device—an LCD. The project was tagged with its title "LCD-Bildschirm". In the taxonomies we would place the tag similar to "Rätsel": As a tag that is "identifying what it is" and meant for "future retrieval". Both tags are "object centric", they refer to the object not to the action i. e. to describe what the object does. From another perspective namely the perspective on the cognitive process (what did the children and young people think when they tagged?) these tags are very different. "Rätsel" describes the project in its context: the story and the part it played within the story. "LCD-Bildschirm" on the other hand is a technical description where the object is taken out of its context. The author indicated "it is exactly the description for what it is". The same author declared in the interview that she did not find her project back with the tag she used. For the next time, she wanted to change her strategy and "add more tags".

> Äh. Womit das irgendwie zusammenhängt oder was es macht. Hat aber nicht ganz geklappt zumindest als ich das mal irgendwo gesucht habe. Habe es nicht wiedergefunden. Aus welchen Gründen auch immer... Also wiedergefunden habe ich ja meins nicht mehr. Weil ich vergessen habe, was ich da eingetippt habe.... Ich könnte mir vorstellen, da mehr Tags rein zu packen und würde das eventuell gut finden, dass die meistbenutzten, die zehn meistbenutzten stehen da und dann klickst du rauf und danach zeigt er die entsprechenden Projekte dazu an. So ne Art Top 100 Liste.
> [schoolgirl in the interview describing why she chose a tag]

> Well. What it relates to or what it does. Did not work out though, at least when I was searching, I did not find it back. No idea why... I did not find mine back. Because I forgot what I typed in... I could imagine using more tags and I would like to see the ten most used tags and then you click on it

> and it shows you the project. A kind of top 100 list.
> [schoolgirl in the interview describing why she chose a tag—translation]

A project that was tagged with an *add several tags strategy* was the Glowing In The Dark Glove. Basically it emitted light according to the light of the environment. The project was tagged with "Glove", "LED", "ligh" and "sensor". We notice one problem known from tagging systems research (see section 5.1.3 for comparison) right away: misspelling. "Ligh" obviously is a mistake and stands for "light". It may confuse users as they will not find the project when looking for the tag "light". We also suspect "light" and "sensor" was meant to be "light-sensor" instead. We already discussed the difficulties of using delimiters in our analysis of the YouTube system (cf. section 5.1.5). The tags are from the same nature as in the projects, we mentioned before. Some tags describe purely technical parts e. g. "LED" or "Sensor".

The tagging on the block level was done differently. Or, at least we can say that the tags that were added are all at the same level. Children and young people tagged the "On" block with "Auto" (Car), "LED", "Motor". The "Delay" block had an example tag of "Länge der Auszeit" (Length of the break). Opposed to the project tags, we do not find tags in the examples that refer to the object or the object's role in the story. We assumed that we would find a similar range of tags, also something like "enterCode" from the children and young people who published the "Rätsel" story, but that is obviously not the case.

We worked with the interview data as we did with the transcripts of the learning experiences we discussed in the previous section.

In the interview with taggers we also showed the tagged projects to them and the "Rätsel" tag worked fine for the children and young people that took part in the workshop. They remembered the story right away. For children and young people that did not take part, the tag was not that understandable. In the usability tests (see section 7.1 for details) "Rätsel" was read and commented with quotes like "Hä? Rätsel was soll das denn?" (German for "Riddle? What is this supposed to mean?").

Another aspect we learned from interviewing the taggers was that they were aware that the tags they gave were visible for other users and the other users have to understand them and make sense of them. The interviewees also asked for a permission system (e. g. to have some tags only visible for the user who gave the tag to something).

When we relate to the categories we discussed in our concepts, we can find evidences for familiarity, personalization and learning or reflection on the tagging process (see table 7.2 for illustration).

Phenomenon	Quotes
Familiarity and Associations	
World of everyday life	"Ah. Something like SchülerVZ." "Interviewer: And you know tagging from the Internet. Schoolgirl: Yo. Interviewer: And you use a system? Schoolgirl: YouTube."

Unfamiliar	"You didn't really know what varible, variab [stutters] variable is. But on and off you know, you can understand."
Appropriations and Contextualizing	
Personalize	"But I would prefer that the one who submitted it [the project] can make one [tag] for everybody and that everybody can make her own [tag]. And oneself can change it afterwards but only for oneself.
Learn and Design Process	
Reflection and interpretation	"Driving, LED? Ah, I see. 'Onfor' is better for me."
	"I would stick with 'on', because we needed it [the block] for LEDs and for motors."
	"I could imagine using more tags and I would like to see the ten most used tags."
Triggering memory	"Atlantis? Ah yes. The explorer and Atlantis" (refers to the story the children and young people invented in which an explorer tried to find Atlantis).
Influence on the design	"Interviewer: If you had looked before and there was already a blinking bag would that change your idea? Interviewee: I think so, because it would be the same thing, nothing special."
Intersubjectivity	"If you tagged it yourself, you of course know, but if somebody looks at it later, then she think she can only use a motor. Or something like this."

Table 7.2: Examples of quotes from the tagging interview

Categorization of the Results

The concept we had for using tagging in Amici consisted of six different objectives and hypotheses. We assumed that tagging could make support easier and more context-depended, that tagging changes the design process and idea building of the children and young people, that tagging is familiar, that tagging has a zeug character and allows multiple interpretations, that tagging helps to design a place and that tagging can help to trigger memory and reflection. When looking at the results of the interviews and the exemplary quotes and codes (cf. table 7.2) we find indications that support some hypotheses while others are not supported.

In our conception of tagging in Amici we suspected the users would use projects and tags for support—the interviewees rather browsed projects and tags out of curiosity. They felt knowing about already existing projects would change their design process, because

they wanted to do something new and unique. When there was already a similar project, they could imagine ending up with working on another idea.

Our interviewees found familiarities in the software and some connected tagging to SNS they knew. This is an indication for the familiarity of tagging that is further supported by having some interviewees referring to Internet forums and suchlike they use in their leisure time. The commands written on the blocks were perceived differently. Some were very understandable (like on and off) others (variable or method) weren't familiar and understandable. Each interviewee mentioned at least one block that was unclear without tags.

The interviewees were keen on using tags for appropriations and individualizations of the software—for example they wanted to have their own private tags on blocks to remember how they used a block. The wish to make their own private place can be seen as a reference to build their own place. However, we assumed beforehand that the shared place would be more important to the children and young people.

The more advanced taggers reflected through the tagging process, but not about the projects as we suspected, but about their tagging strategy. Partly the zeug character of the tags was apparent. When the interviewees reflected upon how to call or tag a block they referred to what use they had in mind and what they wanted to do with it.

Tagging triggered the memory of the participants, the projects and ideas from far away workshops came back when they read a tag.

The interviewees were well aware of other users in the system and they reflected on how the others would interpret and understand the tags.

7.4.1 Tag-based Recommendations

Our assumption that especially children and young people have a problem with big numbers of items and their visualization was confirmed by the usability tests (cf. section 7.1). By the time of the tests the database held eighty-two users. We consider it as necessary to show only a reduced set of users or projects in the visualization.

The test persons also expressed that they would need very similar projects and users to help them with their special programming problems. They answered this way when we asked whether they could imagine asking another user of the system for support.

The most important question of our tag-based recommendations is whether the actual users receive recommendations helpful and somehow understandable to them. At the time of the evaluation we only had eight projects in our database (middle of 2007), a number that is too low to really test the recommender. This problem is a common problem in collaborative filtering often called *cold-start problem* [Schein et al., 2002]. Usually the cold-start problem refers to the situation of a system in which the content is not yet rated and therefore the recommendation engine cannot recommend. In our case it refers to the problem that we do not have enough content in the database yet.

Even with an only slightly greater number of projects in the database (n=19) we received usable results from the recommender (see table 7.3 for two examples). The power of the recommender—to identify co-tags—does not work in the small data set,

the calculated co-tags are the tags used on the project but we still can use the results. When looking at the example project Spooky (a bag with a lot of LEDs that could display different blink patterns), you can see that Spooky shares a tag with all the similar projects the recommender calculated, namely the tag "LED". The recommender does more than this simple matching. The similarity values (calculated with the cosine similarity) show that the project T-shirt for example is more similar to Spooky than the Glowing In The Dark Glove. This is, because of the description tags that refer to sensors and actuators used in the project. Spooky and the T-shirt use the MOTORCONNECTIONTAG and SENSORCONNECTIONTAG—the tags that represent inputs and output pins. They are more similar, because they use the same inputs and outputs.

Table 7.3: Calculated similarities

Name	Tags	Similarity
Drachenauto	rätsel, fahren	
Schatz	fahren	Drachenauto & Schatz: 0.7745966692414834
piratenauto	fahren, auto	Drachenauto & piratenauto: 0.5773502691896258
Spooky	LED, Handtasche	
T-shirt	LED, display, Shirt	Spooky & T-shirt: 0.6123724356957946
WM-shirt	blink, LED	Spooky & WM-shirt: 0.5303300858899106
Backpack	LED, phone, light, bag, sensor	Spooky & Backpack: 0.5
Glowing In The Dark Glove	ligh LED, Glove, sensor	Spooky & Glowing In The Dark Glove: 0.2886751345948129

7.4.2 Usage Statistics

Our database consisted of 126 users, fifty-five tags and thirty projects (at 17 June 2008). The number of project increased from the eight projects in 2007. People used both the web-client as well as the software-client to upload, view and save projects. The tag "LED" was the most popular tag with five projects, "fahren" was used three times to tag projects. Ten projects were tagged with more than one tag. Blocks were tagged with twenty-one distinct tags.

7.5 Conclusion and Summary of the Evaluation Results

From the usability testing we can summarize that the children and young people first of all understood the concept of block programming as well as of tag-based programming and were able to use the software. They understood the tagging concept, even when they did not know it before. Some indicated the concept was familiar for them, because they used web services e. g. YouTube where tags are also used. As visualization, they clearly preferred to see the artifact (our project-centric view) opposed to other users or tags. We cannot conclude from the usability tests that adding a tagcloud to the blocks improves

understanding or usability of the programming, because measuring the task-completion time did not lead to significant results. However, some children and young people clearly mentioned they chose a certain block for programming because of the tags. Rather than because of the representation the designer had chosen.

Taggging provided the recommender with data to base similarity calculations on. Although largely untested, we received results from the recommender that are understandable and make sense—at least for the author. On the other hand the tags were used by children and young people that did not like to read longer descriptions to grasp the idea behind a project quickly.

When we compared our learning environment to the learning environment from our vignette, we can confirm our hypotheses to some extent. Children and young people using the EduWear construction kit did make more connections to their life-world, the world of popular culture and the world of work compared to the participants of the robotics workshop. They perceived the material as flexible and programming as something astonishingly easy. Still, we cannot clearly say what factors lead to the results. The participants were different and also workshop concept and tutors were not the same. How much impact for example the software had, and how much impact the way the tutor explained programming had, is not clear. This is why we decided for a detailed qualitative look on single individual's experiences to see how they felt about the different factors.

Tagging was perceived as a possibility for future retrieval for the individual as well as for others. We noticed that the tags the children and young people used often referred to objects instead of e. g. actions as well. Tagging was used differently, depending on the situation: On the level of projects, we found tags that really described the object's role in a story or idea, while on the block level the description was more direct, more technical.

We could support some of our hypotheses we had connected to tagging. Tagging (or rather sharing of projects) seems to influence the design process by making the children and young people aware of what is already there. It is a familiar technology and it is connected to the world of everyday life and the world of leisure time. Tagging triggers memory and tags have a zeug character. The participants wished to make the programming language their own place and could imagine using tagging to support the process. The other users in the tagging system were perceived and the interviewees had them in mind when thinking about their tags but having their own place was more important to them than building a shared place to socialize.

Chapter 8

Results and Conclusion: Reconsidering Our Work

8.1 Summary

We started this thesis with a vignette, a short scenario. As a reminder, here is the vignette again:

> *Lena, Tobi and Marc have finished building their robot with the help of the manual. Lena is a bit bored; she would have preferred to build something that looks nicer. She did not like the fighting robots from the tutorial. They actually tried to build something else, but the time was short and they were not allowed to glue feathers to the brick. Now the kids are in the process of programming the robot. They want it to go forward till it drives against a wall and then go back. The task seems easy enough, so they start the programming software. "Where is driving? There is 'Action' and 'Variables' but no 'Driving'?" wonders Marc. "Here are also some programs for the Acrobot. Wow, there is one that is called 'dodge' that is exactly what we want." "Does it say more?" asks Lena. "No, only 'dodge'". They try out the program but their robot behaves strangely and falls off the table. Tobi starts to repair it. A tutor explains Marc and Lena that an Acrobot program won't work on their model. They will have to make their own 'dodge program'.*
>
> *"Is there an example how to do it?" asks Lena "Can you show me?" The kids do not understand right away how to create their program. Instead of trying to find a solution they get distracted. "Oh, here it says 'melody'" says Lena. "OK, then let's play a melody instead" answers Marc.*
>
> *Tobi finishes repairing the robot and tries to understand the program the others made. It is not clear to him why they used 'melody' now. Well, he will then stick to the tinkering side and leave programming to the others.*

"Is there an example how to do it?" asks Lena "Can you show me?" The kids do not understand right away how to create their program but instead of trying to find a solution they get distracted. "Oh, here it says 'melody'" says Lena. "OK, then let's play a melody instead" answers Marc.

Tobi finishes repairing the robot and tries to understand the program the others made. It is not clear to him why they used "melody" now. Well, he will then stick to the tinkering side and leave programming to the others.

We used the vignette to identify the problems and their common causes we wanted to work on in this thesis. The children are discouraged, because they cannot build their own artifacts, they have problems understanding representations in the software, and they cannot exchange their ideas properly. What connects these problems is, in our understanding, a missing link to the children and young people's own world and a shared meaning. To summarize: The learning environment they act in is not connected to their context.

After defining the problems we went on clarifying what we mean by context, a term that recently appeared often in interaction design and human computer interaction, but with a lot of different meanings. We introduced a notion of context inspired by researchers such as Dourish, who see context as something dynamic and draw on a phenomenological background. We briefly introduced the background and the implications for interaction design drawing on Dourish's embodied interaction framework. We brought in terms and concepts from the underlying phenomenology, namely (amongst others) *life-world*, *provinces of meaning*, *zeug* and *fields of possibilities*. We then discussed—in connection to Dourish's framework—implications and concepts for the design of digital artifacts including (to recapitulate the most important ones) *familiarities*, *appropriations* and *flexibility*.

With this notion of context and its implications for the design of digital artifacts in mind we started to explore constructionist learning theory as well as learning environments that aim for implementing the approach. We showed that the learning theory goes together quite well with our notion of context, though the creation of a shared context and meaning is a topic that is only recently discussed explicitly in constructionism. This is manifested in additions to Papert's constructionism, e. g. in *constructionist communities* or *distributed constructionism*. We also showed that the social context of the learner is a topic Papert has implicitly dealt with.

Bringing together the contextual perspective with constructionist theory, we introduced a concept for integrating constructionism into context. In the concept, we discussed eleven "design principles" namely: "Start designs from familiarities", "Develop in a participatory design process", "Create a shared context", "Create places", "Allow the communities to create and design their own place", "Consider different aspects of the shared context", "Bring in contextual information in the environment", "Make the programming environment a place and draw on familiarities", "Design for wide walls", "Make something happen in the physical world" and, finally, "Carefully adjust between flexibility and familiarity".

To show that a constructionist learning environment designed with these principles in mind can connect to the users' contexts closely, we decided to study such an environment. To implement the concept, we started by identifying suitable application domains. We particularly looked at two (potentially) suitable technologies in more detail, namely social software focusing on tagging and smart textiles. We could show that both technologies are familiar to the young users, allow to create the users' own places and allow to embed one's context. When trying to approach tagging from a phenomenologists' perspective, we encountered difficulties and contradictions, because in tagging research another perspective—a representational approach—is predominant. However, on the level of implications for the design we could find similarities—research indicates that tagging is flexible, allows appropriations and multiple meanings.

With the EduWear construction kit, we gave an example of a prototype that implements a constructionist learning environment based on our concept and draws on the familiar technologies we introduced. We gave an overview over the design process and the development to demonstrate the use of participatory design methods as a possibility to get an insight into how children and young people work with the materials. We also started to add a new kind of programming environment that took up practices children and young people use on the Internet, especially tagging.

We evaluated the construction kit by deploying different methods, observations of users interacting with the prototypes, as well as a more formal usability testing.

This chapter aims at summarizing as well as discussing our results, pointing out where we think the contribution of this work is and raise questions for future research.

8.1.1 Reflection on the Methods

We started by arguing from a theoretical perspective, clarifying the problems we wanted to research on with our vignette. The vignette proved helpful (at least for the author) to always have the basic problem in mind.

In the construction part of this thesis, we worked on deploying different design methods. Working in a participatory design process inspired by Druin's methods proved valuable, we could gain a starting point for our "imagination", the first step of the design process in Maeda's paradigm. In our research we learned most from observing children and young people using the prototypes and also from asking them how they felt about it in combination with the observations.

When coming to the evaluation we saw ourselves confronted with the following problem: From the digital media perspective a formal evaluation is required: comparing a system to a baseline and showing that it performs better or worse than the baseline system. From the educational perspective on the other hand such a study makes sense only with a huge number of participants to factor out everything that influences a learning process and is not the system—that is, the whole context. Because it is the context we are interested in, such methods do not fit well. Furthermore, a quantitative study does not give an insight into what people experience during the learning process. In this thesis we addressed the problem by deploying usability tests where they seemed

feasible—comparing the software with tagging possibilities to the software without, for example. To get an insight into the process the individual learner experienced we looked closer at a few examples.

The formal usability testing gave some insight into how the concept (of tagging) was used and understood, but only in combination with the qualitative data: the children and young people's utterances during the tests. We could also detect some real usability problems (e. g. forcing the user to double click when she thought only one click was sufficient) that we did not discuss in detail. The quantitative measures (e. g. how long the children and young people needed to perform a task) did not bring interesting results.

Interviewing and observing children and young people in their real environments—the workshops—proved to be the method we found most fruitful to approach our hypotheses. Especially the interviews were interesting, as the children and young people often had a quite clear opinion why they did not like or use parts of the construction kit.

In this thesis, we decided to interpret the constructionist learning environment as a unit, factoring out only pedagogic considerations. Even trying to factor out the pedagogical setting proved difficult. There is an interrelatedness of learners, the setting, the materials they use (including the programming environment) and the constructed artifact that they create. The context in which the creation takes place does matter. This is why we stay very broad in this thesis. As an alternative one could have concentrated on a single component (i. e. the programming environment) of the learning environment and research on the impacts it has. But as this would have put it out of its context we decided not to work that way.

8.2 Results and Conclusions

8.2.1 General Results and Insights

As general results, we could show that tagging, as well as smart textiles, allow to bring in one's context into the learning environment in the sense that both technologies are strongly connected to the children and young people's life-world. We showed this by arguing from a theoretical perspective as well as by discussing the example projects the children and young people made during the design workshops. And by discussing the virtual representations of the projects in the tag-based software. We found out that children and young people use the flexible technology tagging as well as visualization in the software, and even interacting with objects in the real-world in a rather *object-centric* way.

Tags children and young people used describe what the object is and range from technical descriptions (i. e. name of parts) to description of the role in the "story" an object has. They use tagging to integrate their context; the context here refers to the current concept of the project they are working on. The level of what the tags describe is different, also depending on the situation which tagging took place in. Tagging projects and tagging blocks made a difference for the children and young people. We do not know why, but we can suspect that they see the blocks as elements of the programming

language (which they of course are) and therefore search for a tag, a description that is more general, fitting for more situations opposed to tagging their personal projects. The level of abstraction would then be higher when tagging on the block level.

We also gained an insight into how children and young people used tagging in the special environment we created and introduced. Tags referred to objects rather than to actions. Not what the object did mattered, but what role it represented. Within the software children and young people preferred to see these objects rather than the users who made them or descriptions and tags of the objects. This corresponds to the real-world scenario where children and young people started interactions with their peers rather by referring to the object than by directly communicating to the person.

Tagging triggered the memory of the children and young people and it also influenced their design process by making them aware of what is already there.

The children and young people who learned with our learning environment made more connections to their real-world and to the world of work. They felt less limited through the material and perceived the programming as something easier than they expected (all compared to a vignette-like learning environment using Lego Mindstorms as technology). We found indicators that the learning experiences in the workshop situation clearly broadened the field of possibilities the children and young people had.

8.2.2 Implications for Constructionism in Context

We also received some feedback in reference to our concept in which we proposed to design for both familiarity and flexibility. We also mentioned that adjustment between both is important. Our experiences with the prototypes confirmed the assumption we made in the concept. In the implementation, it was difficult to combine both design aspects. By observing the young users interacting with different iterations of the prototypes, we found out that both attributes can be contradictory. Familiarities can also discourage children and young people, when the associations with the familiarity are ill fitting or negative. We had the problem with our first prototypes of the textile patches that were too “toyish”. Great flexibility on the other hand can make the usage harder, as the users cannot draw on familiarities.

For our concept, familiarities and flexibility and their differences and relations should be defined in more detail. The associations connected to the familiarity not only influenced how the children and young people worked with the material, but also how their space of possibility broadened.

To develop in a participatory design process was very important for us. The considerations on flexibility, for example, were possible only because we were able to observe the users’ interactions with the prototypes and asked for their impressions in different iterations of the development.

Through placing the development into the workshop setting, we automatically created shared places for children and young people. Children and young people that took part in the same workshop could remember projects and their meaning by using the tagged project. Participants of other workshops sometimes had problems grasping the

idea behind a project, but were still interested and tried to figure out what the project was about. By taking part in another workshop, they shared the context partly, knowing about different elements and words and terms used during a workshop.

The children and young people we worked with only occasionally used the web-based part of Amici outside the workshops. Even within the workshops using the web-based elements was sometimes a problem. Some tutors did not want the children and young people to have access to the Internet all the time, because the youngsters would start to chat and get distracted. In other scenarios there was no technical infrastructure to provide an Internet connection.

We showed with an example of a huge tagging system, namely YouTube, how tagging can be used to create and design the users' own places. For Amici, we can only say that during the observed use, tags were used as navigational structure and browsing through different projects was an activity the children and young people liked. Through this, tags added a social component to the programming environment. Therefore with Amici we made a step towards a programming environment that is designed by users and acts as a third place.

What we could not show was the building of a common vocabulary, because the amount of projects and users we gathered was too small for data analysis.

For a more general conclusion, we can say that clothing is a good medium to draw on familiarities in a pedagogic environment, because clothing makes it easy for children and young people to integrate their context (how they use and make meaning of clothing) into the learning scenario.

Tagging is a technology that can be used to integrate one's context and even reflect on strategies how to do so. The tag's meaning is dependent on the current context, so it may be hard to understand for people who did not share the context. However, it can help to get a glimpse, a mutual understanding of the other's context. We conclude that tagging can be very valuable for designing (e-)learning environments. The way children and young people use tagging strongly depends on the system's design. We noticed the different use of tagging in Amici depending on whether children and young people tagged blocks or projects.

8.3 Contribution

In this work, we developed a concept founded on theory to allow users to integrate their context into constructionist learning environments. Many considerations we found in interaction design were part of the design decisions of constructionist learning environments (e. g. designing based on familiarities, as Scratch does with taking up the topic of music videos for learning programming). With our concept, we add to the design of constructionist learning environments by making the importance of context explicit and summarize as well as apply the implications. We rather see it as our contribution to add a contextual perspective to the design of constructionist learning environments than to give explicit recommendations *how* to design. Within the concept, we brought together

current directions in interaction design and their notion of context with a constructionist perspective and derived some design implications.

We argue that this broad concept can be used to design constructionist learning environments with various technologies that fulfill the criteria. In this way it is generalizable. We showed that the implementation is possible with choosing the example of tagging and smart textiles, because the application domains seemed suitable. We developed a prototype using the two novel techniques to integrate one's context to demonstrate that the concept works.

By using smart textiles in a learning scenario we present an approach that is innovative, because new and smart materials are seldom used in connection to learning. Furthermore, we showed that the approach works in the scenario as we presented it with our workshops. Up to now, there is only Berglin [Berglin, 2005] and Buechley et al.'s (cf. [Buechley and Eisenberg, 2007]) work in the domain of smart textiles in education. While Berglin uses smart textiles to produce ready-made learning "toys", Buechley's approach is similar to ours: to use the smart textiles as construction material. In difference to Buechley, we created a kit that explores the possibilities of industrial textile techniques e. g. knitting and weaving on large scale machines, to produce the parts of the kits. We showed that this is possible: In the third iteration of the EduWear construction kit we produced a patch that can be put together by children and young people themselves. Through this design, producing the patches requires only very little manual labor. Furthermore, the patches are produced in an entirely textile way. Although such a production is theoretically possible in a larger scale, we decided not to go into a commercial production (that would still be extremely expensive, because of the material costs and a textile PCB cannot be produced in a larger scale yet) and recommend using Buechley et al.'s LilyPads [Buechley et al., 2008] instead when using smart textiles in a constructionist learning environment. The LilyPads are fully compatible with our software Amici. Amici is open source and therefore can be continued and adapted by everybody. To be able to produce construction kits for smart textiles in a larger scale is important to make them available for a wider audience.

We also studied the use of the EduWear construction kit under a special perspective, namely the contextual perspective. We showed how the materials mattered concerning associations and relations the children and young people in our case study made.

Amici as a programming environment brings together visual programming with the block metaphor and social software, especially tagging. Up to now, we find tagging only rarely integrated into programming environments (for children and young people) and if so, as in the example of Scratch, tagging serves as tool when projects are uploaded and shared in a web-based sharing platform. Amici goes one step further by integrating tagging into the programming process itself (by tagging blocks) and by integrating the project sharing features into the environment. Tagging in Amici also serves as a basis to recommend projects and support based on tags, which is a novel approach in the learning domain. We could show that tagging in the programming language can be helpful for the young users.

8.4 Future Directions and Vision

There are many directions we only briefly introduced in this work, as we researched on the different components of the construction kit. We could imagine continuing the research in a pedagogical direction, in a technical one (either in the smart textiles or in the tagging domain), but also to continue research on the conceptual level.

Smart textiles—as we used them in our work—are a technology that is fascinating, because of its familiarity to the cultural practices of clothing and fashion. Robotics on the other hand has a different fascination: to create "artificial life" and autonomous creatures. We limited ourselves, discussing only a specific question, namely: How can we connect to the children and young people's context with our learning environment? That does not answer the question what impact such a connection (in our case the media smart textiles and tagging) has on what the learners really learn during the process and also how the learning process affects their future life.

To answer such questions, we could imagine a large-scale study once again deploying a method triangulation. On the one hand, a quantitative study with many participants where robotics classes, smart textiles classes and—as a control group—conventional computer science classes are compared concerning the models children and young people build on how technology and algorithms work. When we aim at showing that learners learn "more" or "better" about technology, programming and algorithms in a constructionist and contextual environment, we have to compare the environment to a conventional computer science class. On the other hand, a long-term study with only a few participants could help to gather further insights. Especially interesting is how the early and creative use of technology influences the participants' future life and development depending on their context and what factors of their context matter in this development.

From a pedagogical perspective it could be interesting to study the learning process that takes place through tagging. We assume tagging is a tool to reflect on projects and thoughts and externalize the reflections, and to integrate one's context that way. At the same time a shared meaning creation process between a group of several users takes place whenever tagging is done. Concerning the question of how and if shared meaning arises through tagging, a larger database would be helpful. We went online in 2006 and the number of projects, users and tags steadily grew. For a further evaluation in that respect, we suggest to use the database in several workshops and make publishing and sharing a part of the workshop's concept. For further research on how to ease navigation in a large data set, and to continue work on the recommending engine, a larger database would be necessary, too.

The impact the interface or interaction with the system has on how people use tagging is a question that arose in tagging research. It could also be important for the design of learning environments. A goal in designing learning environments may be to design the tagging process in a way that fosters the reflection process. Or—as in our case—to integrate one's context. We saw in our system that the usage patterns on tagging changed, even in the same system, depending on what the tags were used for. Continuing re-

search in that direction could lead to a further classification of tagging systems especially for learning purposes.

We also raised some questions that are important questions in smart textiles research at the moment. Connecting electronics to textiles is one topic. Developing new ways to manufacture smart textile applications is another. We made some experiments in this direction; coming up with the idea to produce the textile parts and let the user put together textiles and electronic components. Such an approach is possible in the learning scenario, but may not be in other domains.

Within the DiMeB research group we offer workshops for beginners and allow children and young people that are interested to scale up and move to more advanced projects by taking part in the expert group. As a vision, the author would like to see such workshops and groups—like in the computer clubhouse approach—spread out, so that every child has the possibility to discover technology in a way meaningful for her and—when she is interested—advancing to more complex projects. The OLPC vision (that we mentioned in section 3.8.4 goes in a similar direction. Smart textiles could be one of the materials available to the children and young people, connecting to their world, culture and tradition as well as to their experiences in crafts, while tagging could be a technique to combine materials and domains. As our contribution, we hope to make designers of such projects and technology aware of the importance the context of the individual has to make interaction with technologies meaningful. In such a large-scale world-wide scenario a research field could emerge that considers context and studies which context factors matter for the learning experience.

Bibliography

[Ackerman, 1996] Ackerman, E. (1996). Perspective–taking and object construction: Two keys to learning. In Kafai, Y. and Resnick, M., editors, *Constructionism in Practice: Designing, Thinking and Learning in a Digital World*, pages 25–35. Lawrence Erlbaum, NJ.

[Ackerman, 2002] Ackerman, E. (2002). Piaget's constructivism, Papert's constructionism: What's the difference? http://learning.media.mit.edu/content/publications/EA.Piaget _ Papert.pdf, Last visit 05.08.2006.

[Backhaus et al., 1989] Backhaus, K., Richson, B., Plinke, W., Schuchard-Ficher, C., and Weiber, R. (1989). Multivariate Analysemethoden. Eine anwendungsorientierte Einführung.

[Baurley, 2004] Baurley, S. (2004). Interactive and experiential design in smart textile products and applications. *Personal Ubiquitous Comput.*, 8(3-4):274–281.

[Begel, 1996] Begel, A. (1996). Logoblocks: A graphical programming language for interacting with the world. Master's thesis, S.B. Thesis, MIT Department of Electrical Engineering and Computer Science.

[Begelman et al., 2006] Begelman, G., Keller, P., and Smadja, F. (2006). Automated tag clustering: Improving search and exploration in the tag space. In *Collaborative Web Tagging Workshop at WWW2006, Edinburgh, Scotland*.

[Berglin, 2004] Berglin, L. (2004). Wanted - textile mobile device. In *Proceedings of VIII International Conference IMTEX 2004 (Interactive Textiles), Lodz, Poland*.

[Berglin, 2005] Berglin, L. (2005). Spookies: combining smart materials and information technology in an interactive toy. In *IDC '05: Proceeding of the 2005 conference on Interaction design and children*, pages 17–23, New York, NY, USA. ACM Press.

[Berzowska, 2005] Berzowska, J. (2005). Electronic textiles: Wearable computers, reactive fashion, and soft computation. *Textile*, 3(1):2–19.

[Bielenberg and Zacher, 2005] Bielenberg, K. and Zacher, M. (2005). Groups in social software: Utilizing tagging to integrate individual contexts for social navigation. Master's thesis, Department of Mathematics and Computer Science, University of Bremen.

[Bilal, 2000] Bilal, D. (2000). Children's use of the yahooligans! web search engine: I. cognitive, physical, and affective behaviors on fact-based search tasks. *Journal of the American Society for Information Science*, 51(7):646–665.

[Boudourides, 2003] Boudourides, M. A. (2003). Constructivism, education, science, and technology. *Canadian Journal of Learning and Technology*, 29(3).

[Boyd, 2007] Boyd, D. (2007). *Why Youth (Heart) Social Network Sites: The Role of Networked Publics in Teenage Social Life*. Buckingham, David (ed.),Cambridge, MA: MIT Press.

[Bruckman, 1997] Bruckman, A. (1997). *MOOSE Crossing: Construction, Community, and Learning in a Networked Virtual World for Kids*. PhD thesis, MIT Media Lab.

[Bruckman, 1998] Bruckman, A. (1998). Community support for constructionist learning. *Comput. Supported Coop. Work*, 7(1-2):47–86.

[Bruckman et al., 2001] Bruckman, A., Druin, A., Inkpen, K., and Preece, J. (2001). The children's challenge: new technologies to support co-located and distributed collaboration. report on the cscw 2000 panel. *SIGCHI Bull.: suppl. interactions*, 2001:6–ff.

[Bruckman and Resnick, 1995] Bruckman, A. and Resnick, M. (1995). The MediaMOO project: Constructionism and professional community. *Convergence*, 1(1).

[Bruckman and Resnick, 1996] Bruckman, A. and Resnick, M. (1996). The MediaMOO project: Constructionism and professional community. In Kafai, Y. B. and Resnick, M., editors, *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*, pages 207–221. Lawrence Erlbaum Associates.

[Bryant and Charmaz, 2008] Bryant, A. and Charmaz, K., editors (2008). *The SAGE handbook of grounded theory*. Sage Publications.

[Buechley, 2006] Buechley, L. (2006). A construction kit for electronic textiles. In *Proceedings of IEEE International Symposium on Wearable Computers (ISCW), Montreux, Switzerland*.

[Buechley and Eisenberg, 2007] Buechley, L. and Eisenberg, M. (2007). Fabric pcbs, electronic sequins, and socket buttons: techniques for e-textile craft. *Personal and Ubiquitous Computing*.

[Buechley et al., 2008] Buechley, L., Eisenberg, M., Catchen, J., and Crockett, A. (2008). The Lilypad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 423–432, New York, NY, USA. ACM.

[Buechley et al., 2007] Buechley, L., Eisenberg, M., and Elumeze, N. (2007). Towards a curriculum for electronic textiles in the high school classroom. *SIGCSE Bull.*, 39(3):28–32.

[Buechley et al., 2005] Buechley, L., Elumeze, N., Dodson, C., and Eisenberg, M. (2005). Quilt snaps: A fabric based computational construction kit. In *Proceedings of IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE), Tokushima, Japan, November 2005*.

[Buechley et al., 2006] Buechley, L., Elumeze, N., and Eisenberg, M. (2006). Electronic/computational textiles and children's crafts. In *IDC '06: Proceeding of the 2006 conference on Interaction design and children*, pages 49–56, New York, NY, USA. ACM Press.

[Burke, 2002] Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370.

[Campbell, 2006] Campbell, D. G. (2006). A phenomenological framework for the relationship between the semantic web and user-centered tagging systems. In Furner, J. and Tennis, J. T., editors, *Proceedings 17th Workshop of the American Society for Information Science and Technology Special Interest Group in Classification Research 17, Austin, Texas*.

[Chalmers, 2003] Chalmers, M. (2003). Informatics, architecture and language. In *Designing information spaces: the social navigation approach*, pages 315–341. Springer-Verlag, London, UK.

[Chapman, 2003] Chapman, R. (2003). Pearls of wisdom: Learning in distributed constructionist cooperatives. Phd proposal MIT Media Laboratory.

[Coates, 2005] Coates, T. (2005). An addendum to a definition of social software. PLASTICBAG.ORG. Last visit 25 April 2008.

[Coffin, 2008] Coffin, J. (2008). Robotany and Lichtung: a contribution to phenomenological dialogue. In *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 217–220, New York, NY, USA. ACM.

[Collins et al., 1989] Collins, A., Brown, J., and Newman, S. (1989). *Cognitive apprenticeship: Teaching the craft of reading, writing and mathematics*. Hillsdale, NJ: Erlbuam.

[Craik, 1994] Craik, J. (1994). *The face of fashion*. Routledge, London and New York.

[Crawford, 2002] Crawford, C. (2002). *The Art of Interactive Design: A Euphonious and Illuminating Guide to Building Successful Software*. No Starch Press.

[de Nooy et al., 2005] de Nooy, W., Mrvar, A., and Batagelj, V. (2005). *Exploratory Social Network Analysis with Pajek (Structural Analysis in the Social Sciences)*. Cambridge University Press.

[Dekoli and Mikhak, 2004] Dekoli, M. and Mikhak, B. (2004). Codachrome: a system for creating interactive electronic jewelry for children. In *IDC '04: Proceeding of the 2004 conference on Interaction design and children*, pages 139–140, New York, NY, USA. ACM Press.

[Dey et al., 2001] Dey, A. K., Abowd, G. D., and Salber, D. (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16:97–166.

[DiSessa, 2000] DiSessa, A. (2000). *Changing Minds*. MIT Press.

[Dittert et al., 2008] Dittert, N., Dittmann, K., Grüter, T., Kümmel, A., Osterloh, A., Reichel, M., Schelhowe, H., Volkmann, G., and Zorn, I. (2008). Understanding digital media by constructing intelligent artefacts. Design of a learning environment for children. In *Proceedings of Ed-Media. World Conference on Educational Multimedia, Hypermedia & Telecommunications*, Vienna, Austria. to appear in print.

[Dourish, 2001a] Dourish, P. (2001a). Seeking a foundation for context-aware computing. *Human-Computer Interaction*, 16:229–241.

[Dourish, 2001b] Dourish, P. (2001b). *Where the Action Is: The Foundations of Embodied Interaction*. MIT Press, Cambridge.

[Dourish, 2004] Dourish, P. (2004). What we talk about when we talk about context. *Personal Ubiquitous Comput.*, 8(1):19–30.

[Dourish, 2006a] Dourish, P. (2006a). Re-space-ing place: "place" and "space" ten years on. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 299–308, New York, NY, USA. ACM.

[Dourish, 2006b] Dourish, P. (2006b). Re-space-ing place: "place" and "space" ten years on. In *CSCW '06: Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, pages 299–308, New York, NY, USA. ACM.

[Dourish and Chalmers, 1994] Dourish, P. and Chalmers, M. (1994). Running out of space: Models of information navigation. In *HCI'94, Glasgow*.

[Druin, 1999] Druin, A. (1999). Cooperative inquiry: Developing new technologies for children with children. In *Proceedings of CHI'99*, pages 592–599, New York. ACM, ACM.

[Eco, 1973] Eco, U. (1973). Social life as a sign system. In Robey, D., editor, *Structuralism: An introduction*, pages 57–72. Clarendon, Oxford, UK.

[Edthofer, 2003] Edthofer, C. (2003). Body-horror. Körperinszenierung als Teilaspekt medialer Identität in Musikvideos. www.alexanderhuemer.com/clarissaedthofer. Last visit 25 April 2008.

[Eisenberg, 2005] Eisenberg, M. (2005). The material side of educational technology. *Commun. ACM*, 48(1):51–54.

[Eisenberg et al., 2006] Eisenberg, M., Eisenberg, A., Buechley, L., and Elumeze, N. (2006). Invisibility considered harmful: Revisiting traditional principles of ubiquitous computing in the context of education. In *WMTE '06: Proceedings of the Fourth*

IEEE International Workshop on Wireless, Mobile and Ubiquitous Technology in Education, pages 103–110, Washington, DC, USA. IEEE Computer Society.

[Eisenberg et al., 2003] Eisenberg, M., Eisenberg, A., Hendrix, S., Blauvelt, G., Butter, D., Garcia, J., Lewis, R., and Nielsen, T. (2003). As we may print: new directions in output devices and computational crafts for children. In *IDC '03: Proceeding of the 2003 conference on Interaction design and children*, pages 31–39, New York, NY, USA. ACM Press.

[El-Sherif, 2005] El-Sherif, M. (2005). Integration of fibre optic sensors and sensing networks into textile structures. In Tao, X., editor, *Wearable electronics and photonics*, chapter 6. Woodhead Publishing.

[Elliott, 2007] Elliott, C. M. (2007). Tangible functional programming. In *ICFP '07: Proceedings of the 2007 ACM SIGPLAN international conference on Functional programming*, pages 59–70, New York, NY, USA. ACM.

[Ellwanger, 2007] Ellwanger, M. (2007). Process textile development. Presentation on the 3rd transnational meeting EduWear 2007.

[Evard, 1996] Evard, M. (1996). Communities of designers: Learning through exchanging questions and answers. In Kafai, Y. B. and Resnick, M., editors, *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*, pages 223–239. Lawrence Erlbaum Associates.

[Feld, 1981] Feld, S. L. (1981). The focussed organization of social ties. *The American Journal of Sociology, The University of Chicago Press*, 86(5):1015–1035.

[Fernaeus et al., 2008] Fernaeus, Y., Tholander, J., and Jonsson, M. (2008). Towards a new set of ideals: consequences of the practice turn in tangible interaction. In *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, pages 223–230, New York, NY, USA. ACM.

[Floyd, 2002] Floyd, C. (2002). Developing and embedding autooperational form. In Dittrich, Y., Floyd, C., and Klischewski, R., editors, *Social thinking—software practice*, pages 5–28. MIT Press, Cambridge, Massachusetts, London, England.

[Gehrke, 2008] Gehrke, D. (2008). Emma. Konzeption und Implementierung eines Empfehlungssystems zur Unterstützung von kooperativen Lernplattformen. Master's thesis, University of Bremen, Department of Mathematics and Computer Science.

[Geisler and Burns, 2007] Geisler, G. and Burns, S. (2007). Tagging video: conventions and strategies of the youtube community. In *Proceedings of the 2007 conference on Digital libraries, Vancouver, BC, Canada*.

[Gibson, 1977] Gibson, J. J. (1977). The theory of affordances. In Shaw, R. and Bransford, J., editors, *Perceiving, Acting, and Knowing*.

[Gindling et al., 1995] Gindling, J., Ioannidou, A., Loh, J., and Lokkebo, O. (1995). Legosheets: A rule-based programming, simulation and manipulation environment for the lego programmable brick. In *Proceeding of Visual Languages*, pages 172–179. IEEE Computer Society Press.

[Goldberg et al., 1992] Goldberg, D., Nichols, D., Oki, B. M., and Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12):61–70.

[Golder and Huberman, 2005] Golder, S. and Huberman, B. (2005). The structure of collaborative tagging systems. http://www.citebase.org/cgi-bin/citations?id=oai:arXiv.org:cs/0508082. Last visit 25 April 2008.

[Grueter, 2006] Grueter, T. (2006). Software as an element of a constructionist learning environment using robots as personal-meaningful and engaging artefacts for children. Master's thesis, DiMeB, University of Bremen.

[Guha et al., 2004] Guha, M. L., Druin, A., Chipman, G., Fails, J. A., Simms, S., and Farber, A. (2004). Mixing ideas: a new technique for working with young children as design partners. In *IDC '04: Proceeding of the 2004 conference on Interaction design and children*, pages 35–42, New York, NY, USA. ACM Press.

[Guzdial, 2004] Guzdial, M. (2004). Programming environments for novices. *Computer Science Education Research*.

[Haraway, 1991] Haraway, D. J. (1991). *A Cyborg Manifesto: Science, Technology, and Socialist-Feminism in the Late Twentieth Century*. Free Association Books / Routledge.

[Harrison and Dourish, 1996] Harrison, S. and Dourish, P. (1996). Re-place-ing space: the roles of place and space in collaborative systems. In *CSCW '96: Proceedings of the 1996 ACM conference on Computer supported cooperative work*, pages 67–76, New York, NY, USA. ACM.

[Hebdige, 1979] Hebdige, D. (1979). *Subculture: The Meaning of Style*. Methuan and Co, Ltd, 733 Third Avenue, New York.

[Heidegger, 1962] Heidegger, M. (1962). *Being and time*. Harper & Row, New York, Hagerstown, San Francisco, London. Translation from John Macquarrie & Edward Robinson.

[Heidegger, 1967] Heidegger, M. (1967). *Sein und Zeit*. Max Niemeyer Verlag, 11th edition.

[Honkala, 2006] Honkala, M. (2006). Introduction to shape memory materials. In Mattila, H., editor, *Intelligent textiles and clothing*, pages 85–103. Woodhead Publishing Ltd, Cambridge, England.

[Höök, 2006] Höök, K. (2006). Designing familiar open surfaces. In *NordiCHI '06: Proceedings of the 4th Nordic conference on Human-computer interaction*, pages 242–251, New York, NY, USA. ACM.

[Ingalls et al., 1997] Ingalls, D., Kaehler, T., Maloney, J., Wallace, S., and Kay, A. (1997). Back to the future: the story of squeak, a practical smalltalk written in itself. *SIGPLAN Not.*, 32(10):318–326.

[Ishii and Ullmer, 1997] Ishii, H. and Ullmer, B. (1997). Tangible bits: towards seamless interfaces between people, bits and atoms. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 234–241, New York, NY, USA. ACM.

[Jones et al., 2003] Jones, C., McIver, L., Gibson, L., and Gregor, P. (2003). Experiences obtained from designing with children. In *IDC '03: Proceeding of the 2003 conference on Interaction design and children*, pages 69–74, New York, NY, USA. ACM Press.

[Kafai and Resnick, 1996] Kafai, Y. B. and Resnick, M., editors (1996). *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*. Lawrence Erlbaum Associates.

[Kahn, 1996] Kahn, K. (1996). Toon talk: An animated programming environment for children. *Journal of Visual Languages and Computing*, 7(2):197 – 217.

[Kaser and Lemire, 2007] Kaser, O. and Lemire, D. (2007). Tag-cloud drawing: Algorithms for cloud visualization. http://www.citebase.org/abstract?id=oai:arXiv.org:cs/0703109. Last visit 25 April 2008.

[Kay, 1972] Kay, A. (1972). A personal computer for children of all ages. Draft, Xerox Palo Alto Research Center, Last visit 3 August 2008.

[Kelleher and Pausch, 2005] Kelleher, C. and Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Comput. Surv.*, 37(2):83–137.

[Kirstein et al., 2005] Kirstein, T., Cottet, D., Grzyb, J., and Tröster, G. (2005). Wearable computing systems - electronic textiles. *Wearable Electronics and Photonics*, pages 177–197.

[Kohlhase and Reichel, 2008] Kohlhase, A. and Reichel, M. (2008). Embodied conceptualizations: Social tagging and e-learning. *International Journal of Web-Based Learning and Teaching Technologies*, 3:58–67.

[Konstan and Riedl, 2003] Konstan and Riedl (2003). Collaborative filtering: Supporting social navigation in large, crowded infospaces. In *Designing Information Spaces: The Social Navigation Approach*, pages 43–82. Springer.

[Langeveld, 1984] Langeveld, M. J. (1984). How does the child experience the world of things? *Phenomenology +Pedagogy*, 2(3):215–223.

[Latour, 2007] Latour, B. (2007). *Reassembling the Social: An Introduction to Actor-Network-Theory (Clarendon Lectures in Management Studies)*. Oxford University Press, USA.

[Lave and Wenger, 1991] Lave, J. and Wenger, E. (1991). *Situated Learning : Legitimate Peripheral Participation*. Cambridge University Press.

[Lave and Wenger, 1996] Lave, J. and Wenger, E. (1996). Practice, person, social world. In Daniels, H., editor, *An introduction to Vygotsky*, pages 143–150. Routledge.

[Lee et al., 2005] Lee, S., du Preez, W., and Jones, N. T. (2005). *Fashioning the Future: Tomorrow's Wardrobe*. Thames and Hudson Ltd.

[Lerman, 2007] Lerman, K. (2007). Social browsing & information filtering in social media. *CoRR*, abs/0710.5697. Informal publication.

[Lévi-Strauss, 1968] Lévi-Strauss, C. (1968). *The Savage Mind (Nature of Human Society)*. University Of Chicago Press.

[Maeda, 1999] Maeda, J. (1999). *Design by Numbers*. MIT Press, Cambridge, MA, USA. Foreword By-Paola Antonelli.

[Maeda, 2000] Maeda, J. (2000). *Maeda@Media*. Rizzoli Publications, New York.

[Maloney et al., 2004] Maloney, J., Burd, L., Kafai, Y., Rusk, N., Silverman, B., and Resnick, M. (2004). Scratch: A sneak preview. In *Second International Conference on Creating, Connecting, and Collaborating through Computing.*, pages 104–109.

[Mann, 1998] Mann, S. (1998). Wearable computing as means for personal empowerment. In *Keynote Address for The First International Conference on Wearable Computing, ICWC-98, May 12-13, Fairfax VA*.

[Marinchev, 2006] Marinchev (2006). Practical semantic web. Tagging and tag clouds. *Cybernetics and Information Technologies*, 3.

[Marlow et al., 2006] Marlow, C., Naaman, M., Boyd, D., and Davis, M. (2006). Ht06, tagging paper, taxonomy, flickr, academic article, to read. In *HYPERTEXT '06: Proceedings of the seventeenth conference on Hypertext and hypermedia*, pages 31–40, New York, NY, USA. ACM Press.

[Martin, 2006] Martin, F. (2006). Cricketlogo and support on the internet. Interview with the author 11 October 2006, ZIM Bremen, Cassette recording in possession of author.

[Martin et al., 2000] Martin, F., Mikhak, B., Resnick, M., Silverman, B., and Berg, R. (2000). To mindstorms and beyond: Evolution of a construction kit for magical machines. In J. Hendler, A. D., editor, *Robots For Kids*, chapter 1. Morgan Kaufmann.

[Martin et al., 2004] Martin, T., Jones, M., Edmison, J., Sheikh, T., and Nakad, Z. (2004). Modeling and simulating electronic textile applications. *SIGPLAN Not.*, 39(7):10–19.

[Mathes, 2004] Mathes, A. (2004). Folksonomies - cooperative classification and communication through shared metadata. http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html. Last visit 25 April 2008.

[McNerney, 2000] McNerney, T. S. (2000). Tangible programming bricks:an approach to making programming accessible to everyone. Master's thesis, Program in Media Arts and Sciences,Massachusetts Institute of Technology.

[Medienpädagogischer Forschungsverbund Südwest, 2006] Medienpädagogischer Forschungsverbund Südwest (2006). Jim studie 2006. http://www.mpfs.de/index.php?id=11. Last visit 25 April 2008.

[Medienpädagogischer Forschungsverbund Südwest, 2007] Medienpädagogischer Forschungsverbund Südwest (2007). Jim Studie 2007. http://www.mpfs.de/index.php?id=110. Last visit 25 April 2008.

[Meoli and May-Plumlee, 2002] Meoli, D. and May-Plumlee, T. (2002). Interactive electronic textile development: A review of technologies. *Textile and Apparel, Technology and Management*, 2(2).

[Merleau-Ponty et al., 1968] Merleau-Ponty, M., Lefort, C., and Lingis, A. (1968). *The Visible and the Invisible: Followed by Working Notes*. Northwestern University Press. Translation from Alphonso Lingis.

[Monroy-Hernández, 2007] Monroy-Hernández, A. (2007). ScratchR: sharing user-generated programmable media. In *IDC '07: Proceedings of the 6th international conference on Interaction design and children*, pages 167–168, New York, NY, USA. ACM.

[Monroy-Hernández and Resnick, 2008] Monroy-Hernández, A. and Resnick, M. (2008). Feature empowering kids to create and share programmable media. *interactions*, 15(2):50–53.

[Moran and Dourish, 2001] Moran, T. P. and Dourish, P. (2001). Introduction to this special issue on context-aware computing. *Human-Computer Interaction*, 16:87–95.

[Nielsen, 1997] Nielsen, J. (1997). Usability engineering. In *The Computer Science and Engineering Handbook*, pages 1440–1460.

[Nielsen, 2002] Nielsen, J. (2002). Kids' corner: Website usability for children. http://www.useit.com/alertbox/children.html. Last visit 30 July 2008.

[Norman, 2002] Norman, D. A. (2002). *The Design of Everyday Things*. Basic Books.

[Ohmatex, 2007] Ohmatex (2007). White paper on smart textiles - market overview. www.ohmatex.dk. Last visit 25 April 2008.

[Oldenburg, 1989] Oldenburg, R. (1989). *The great good place*. Paragon House, New York.

[O'Reilly, 2005] O'Reilly, T. (2005). What is Web 2.0. Design patterns and business models for the next generation of software. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html. Last visit 25 April 2008.

[Pahl, 2004] Pahl, C. (2004). Kleidsame Elektronik. http://www.heise.de/tr/artikel/53360. Last visit 25 April 2008.

[Papert, 1976] Papert, S. (1976). Some poetic and social criteria for education design (aim-373). Technical report.

[Papert, 1980] Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books.

[Papert, 2000] Papert, S. (2000). What's the big idea? toward a pedagogy of idea power. *IBM Systems Journal*, 39(3/4):720–729.

[Papert, 2002] Papert, S. (2002). Hard fun. *Article for the Bangor Daily News (Bangor, Maine)*.

[Papert and Harel, 1991] Papert, S. and Harel, I. (1991). Situating constructionism. In Seymour Papert, I. H., editor, *Constructionism*, chapter Introduction. Ablex Publishing: New York, NY.

[Papert and Resnick, 1995] Papert, S. and Resnick, M. (1995). Technological fluency and the representation of knowledge. Proposal to the National Science Foundation, MIT Media Laboratory, Cambridge, MA.

[Papert and Solomon, 1971] Papert, S. and Solomon, C. (1971). Twenty things to do with a computer. AI Memos AIM-248, MIT.

[Peppler and Kafai, 2006] Peppler, K. A. and Kafai, Y. B. (2006). Creative codings: investigating cultural, personal, and epistemological connections in media arts programming. In *ICLS '06: Proceedings of the 7th international conference on Learning sciences*, pages 972–973. International Society of the Learning Sciences.

[Picard, 1997] Picard, R. W. (1997). *Affective computing*. Cambridge, MA: MIT Press.

[Plassmeier, 2007] Plassmeier, S. (2007). Robokids. Konzeption und Implementierung eines Frameworks zur Unterstuetzung von kooperativem Lernen. Master's thesis, Department of Mathematics and Computer Science, University of Bremen.

[Post et al., 2000] Post, E. R., Orth, M., Russo, P. R., and Gershenfeld, N. (2000). E-broidery: design and fabrication of textile-based computing. *IBM Syst. J.*, 39(3-4):840–860.

[Pruitt and Grudin, 2003] Pruitt, J. and Grudin, J. (2003). Personas: practice and theory. In *DUX '03: Proceedings of the 2003 conference on Designing for user experiences*, pages 1–15, New York, NY, USA. ACM.

[Redstroem et al., 2005] Redstroem, M., Redstroem, J., and Maze, R. (2005). *IT + Textiles*. Edita Publishing Oy, Finland.

[Reichel, 2007] Reichel, M. (2007). Meeting at the Titv institute 22 November 2007.

[Reichel et al., 2008] Reichel, M., Osterloh, A., Katterfeldt, E.-S., Schelhowe, H., and Butler, D. (2008). Eduwear: Designing smart textiles for playful learning. In *Proceedings ICICTE International Conference on Information Communication Technologies in Education*, Corfu, Greece. To appear in print.

[Reichel and Wiesner-Steiner, 2006] Reichel, M. and Wiesner-Steiner, A. (2006). Gender inscriptions in robotic courses. In *1st Internationl Conference on Digital Media and Learning 2006 (ICDML 2006)*, pages 61–65, Bangkok.

[Repenning and Ambach, 1996] Repenning, A. and Ambach, J. (1996). Tactile programming: A unified manipulation paradigm supporting program comprehension, composition, and sharing. In *IEEE Symposium on Visual Languages, Boulder, CA*.

[Repenning et al., 1999] Repenning, A., Ioannidou, A., and Phillips, J. (1999). Collaborative use & design of interactive simulations. In *CSCL '99: Proceedings of the 1999 conference on Computer support for collaborative learning*, page 59. International Society of the Learning Sciences.

[Resnick, 1993] Resnick, M. (1993). Behavior construction kits. *Communications of the ACM*, 36(7).

[Resnick, 1994] Resnick, M. (1994). *Turtles, termites, and traffic jams: explorations in massively parallel microworlds*. MIT Press, Cambridge, MA, USA.

[Resnick, 1996] Resnick, M. (1996). Distributed constructionism. In *ICLS '96: Proceedings of the 1996 international conference on Learning sciences*, pages 280–284. International Society of the Learning Sciences.

[Resnick, 2004] Resnick, M. (2004). Edutainment? No thanks. I prefer playful learning. *Associazione Civita Report on Edutainment*.

[Resnick et al., 1996] Resnick, M., Martin, F., R.Sargent, and Silverman, B. (1996). Programmable bricks: Toys to think with. *IBM Systems Journal*, 35(3-4):443–452.

[Resnick and Silverman, 2005] Resnick, M. and Silverman, B. (2005). Some reflections on designing construction kits for kids. In *IDC '05: Proceeding of the 2005 conference on Interaction design and children*, pages 117–122, New York, NY, USA. ACM Press.

[Resnick and Varian, 1997] Resnick, P. and Varian, H. R. (1997). Recommender systems. *Commun. ACM*, 40(3):56–58.

[Rheingold, 2003] Rheingold, H. (2003). *Smart Mobs: The Next Social Revolution*. Basic Books.

[Rivadeneira et al., 2007] Rivadeneira, A. W., Gruen, D. M., Muller, M. J., and Millen, D. R. (2007). Getting our head in the clouds: toward evaluation studies of tagclouds. In *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 995–998, New York, NY, USA. ACM Press.

[Russell, 2006] Russell, T. (2006). cloudalicious: folksonomy over time. In *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 364–364, New York, NY, USA. ACM Press.

[Sargent et al., 1996] Sargent, R.and Resnick, M., Martin, F., and Silverman, B. (1996). Building and learning with programmable bricks. In Kafai, Y. and Resnick, M., editors, *Constructionism in Practice*. Lawrence Erlbaum, Mahwah, NJ.

[Scaife et al., 1997] Scaife, M., Rogers, Y., Aldrich, F., and Davies, M. (1997). Designing for or designing with? Informant design for interactive learning environments. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 343–350, New York, NY, USA. ACM Press.

[Schein et al., 2002] Schein, A., Popescul, A., Ungar, L., and Pennock, D. (2002). Methods and metrics for cold-start recommendations. In *Proceedings of the 25 Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2002)*, pages 253–260.

[Schelhowe, 2007] Schelhowe, H. (2007). *Technologie, Imagination und Lernen*. Waxmann, Münster, New York, München, Berlin.

[Schütz, 1962] Schütz, A. (1962). On multiple realities. In Natanson, M., editor, *A. Schutz: Collected Papers. Vol. I. The Problem of Social Reality*, pages 207–259. Martinus Nijhoff, The Hague.

[Sengers and Gaver, 2006] Sengers, P. and Gaver, B. (2006). Staying open to interpretation: engaging multiple meanings in design and evaluation. In *DIS '06: Proceedings of the 6th ACM conference on Designing Interactive systems*, pages 99–108, New York, NY, USA. ACM Press.

[Seymour, 2008] Seymour, S. (2008). *Fashionable Technology: The Intersection of Design, Fashion, Science and Technology*. Springer, Wien, first edition.

[Shaw, 1996] Shaw, A. (1996). Social constructionism and the inner city: Designing environments for social development and urban renewal. In Kafai, Y. B. and Resnick, M., editors, *Constructionism in Practice: Designing, Thinking, and Learning in a Digital World*, pages 175–206. Lawrence Erlbaum Associates.

[Shook, 2000] Shook, M. L. (2000). Scaffolding children's exploration of motion and mechanism. Master's thesis, Thesis (S.M.)– (S.M.)Massachusetts Institute of Technology, Dept. of Mechanical Engineering.

[Sinha, 2006] Sinha, R. (2006). A social analysis of tagging (or how tagging transforms the solitary browsing experience into a social one). http://www.rashmisinha.com/archives/06_01/social-tagging.html. Last visit 25 April 2008.

[Smith and Trophan, 2005] Smith, C. and Trophan, S. (2005). *Xtreme Fashion*. Prestel Publishing LTD, Munich, Berlin, London, New York.

[Sood et al., 2007] Sood, S., Owsley, S., Hammond, K., and Birnbaum, L. (2007). Tagassist: Automatic tag suggestion for blog posts. *Proceedings of ICWSM 2007*.

[Specia and Motta, 2007] Specia, L. and Motta, E. (2007). Integrating folksonomies with the semantic web. In *The Semantic Web: Research and Applications*, pages 624–639.

[Steiner and Atzmüller, 2006] Steiner, P. M. and Atzmüller, C. (2006). Experimentelle Vignettendesigns in faktoriellen Surveys. *KZfSS Kölner Zeitschrift für Soziologie und Sozialpsychologie*, Volume 58:117 – 146.

[Suchman, 1987] Suchman, L. A. (1987). *Plans and Situated Actions. The problem of human machine communication*. Cambridge University Press, Cambridge (MA).

[Tao, 2005] Tao, X. (2005). Introduction. In Tao, X., editor, *Wearable Electronics and Photonics*, pages 1–12. Woodhead Publishing Limited, Abington Hall, Abington, Cambridge, CB1 6AH, England.

[Tepper, 2003] Tepper, M. (2003). The rise of social software. *netWorker*, 7(3):18–23.

[TeTRInno, 2007] TeTRInno (2007). State of the art in smart textiles and interactive fabrics. http://www.mateo.ntc.zcu.cz/index.php. Last visit 25 April 2008.

[Tholander and Fernaeus, 2004] Tholander, J. and Fernaeus, Y. (2004). Embodied programming with visual and tangible representations. In *SIG-CSCL conference*.

[Turkle, 1997] Turkle, S. (1997). *Life on the Screen: Identity in the Age of the Internet*. Simon & Schuster.

[Turkle and Papert, 1991] Turkle, S. and Papert, S. (1991). Epistemological pluralism and the revaluation of epistemological pluralism and the revaluation of the concrete. In Papert, S. and Harel, I., editors, *Constructionism*. Ablex Publishing: New York, NY.

[Vygotsky, 1978] Vygotsky, L. (1978). *Mind in Society: The development of higher psychological processes*. Harvard University Press, Cambridge, MA.

[Wilker, 2007] Wilker, I. (2007). Social web 101 for nonprofits. http://ianwilker.com/rootslab/2007/07/04/social-web-101/. Last visit 25 April 2008.

[Williamson, 2003] Williamson, B. (2003). The participation of children in the design of new technology. *Futurelab Discussion Paper*.

[Zainzinger, 2007] Zainzinger, J. (2007). Hussein chalayan: fashion's great innovator. blog. http://www.talk2myshirt.com/blog/archives/127, Last visit 26 June 2008.

List of Figures

3.1 Output of the turtle . . . 38
3.2 Programmable bricks as fantasy creatures for a performance. . . . 40
3.3 Program with LogoBlocks in the version for Handy Crickets . . . 41

5.1 Suggestions from Delicious . . . 70
5.2 Personal tagcloud in Delicious . . . 71
5.3 Folksonomy Delicious . . . 72
5.4 Visualizing tag relations with EMMA . . . 75
5.5 Meaning creation with tags of the individual . . . 79
5.6 Meaning creation with tags in social tagging . . . 79
5.7 Tagging systems modeled as tripartite network . . . 82
5.8 Basic data of YouTube's popular videos . . . 84
5.9 YouTube's friendships depending on indicated age . . . 84
5.10 Most common tags in the Make community . . . 85
5.11 Connection through friendships . . . 87
5.12 Connection in a set of random users . . . 88
5.13 A couch with Lumalive by Phillips . . . 95
5.14 Warp and weft yarns in weaving . . . 97
5.15 Students' sketch of a bicycle messenger's brake light. . . . 100
5.16 The Smart Dance clothes react to movement, e. g. dancing or jumping. . . . 101
5.17 The Smart Dance shirts react to the movement with changing light patterns. 102
5.18 The wearer can record and send a visualization of the light patterns to a friend's mobile. . . . 103
5.19 The resulting visualization of the wearer's movement on the horizontal level 103
5.20 Scorpions, an XS Labs project by Berzowska and Mainstone . . . 105
5.21 What does clothing mean to you? . . . 107
5.22 What does clothing mean to you? . . . 107
5.23 Age of the children and young people we asked what clothing meant to them. . . . 108
5.24 Gender of the children and young people we asked what clothing meant to them . . . 108

6.1 Sketches from the future design session . . . 119

6.2 Passing and interacting depending on different models 121
6.3 First prototype 131
6.4 Second prototype 132
6.5 Knotting technique on the left and weaving the components directly into the weave in a color-coded LED fabric on the right. 133
6.6 The third prototype with explanations of the single parts. 133
6.7 Different iterations of the textile PCB and the third prototype. 134
6.8 Gluing technique from Titv 135
6.9 Simple use case of the Amici software. 139
6.10 Zoom level one 141
6.11 Zoom level two 141
6.12 Zoom level three: code in the visual representation 142
6.13 Zoom level four 142
6.14 The user switches an LED on 143
6.15 Block architecture 144
6.16 Tagging on the block level 145
6.17 Publish and tag a project 146
6.18 Overall architecture of the EduWear construction kit 147
6.19 Different views from left to right: user-centric, project-centric, tag-centric . 152

7.1 Task-completion time for the task “switch on an LED” in a view with the tagcloud. 158
7.2 Task-completion time for the task “switch an LED on” in a view without the tagcloud. 159
7.3 Age curve of the participants and their preferred view 159
7.4 Diagram depicting the users’ experiences in programming 160
7.5 Diagram depicting the users’ reasons for choosing projects in the tests. . . 160
7.6 Diagram depicting the users’ reasons for choosing friends in the tests . . . 161
7.7 Age of the participants 164
7.8 On the left: Did you ever open a device (e. g. a radio) to see how it works? On the right: Did you ever talk about technology with your friends? 164
7.9 The sketch of Andy’s group 166
7.10 The robot Andy’s group made during the workshop 167
7.11 Mehmet and his group’s dinosaur sketch 169
7.12 Mehmet and his group’s final project 170
7.13 Age and gender of the participants 173
7.14 On the left: Did you ever open a device (e. g. a radio) to see how it works? On the right: Did you ever talk about technology with your friends? 173
7.15 (Technical) sketches of the sexy bag 174
7.16 The bag as a part of Anne’s project 175
7.17 A sketch of the star shirt 176
7.18 Working on the star shirt 176
7.19 The star shirt project in Amici’s browser-based client 177

7.20 The star shirt's program 178

A.1 A program that uses the setup, the loop and the method block. 218
A.2 Presenting the jacket for the blind on the catwalk 220
A.3 The jacket for the blind's program in its visual representation 221
A.4 The jacket for the blind's program in Amici's browser-based client 223

Appendix A

Two Example Programs

One example program is written by the author to explain, how the visual programming language works, while the other example is from two children that took part in a workshop.

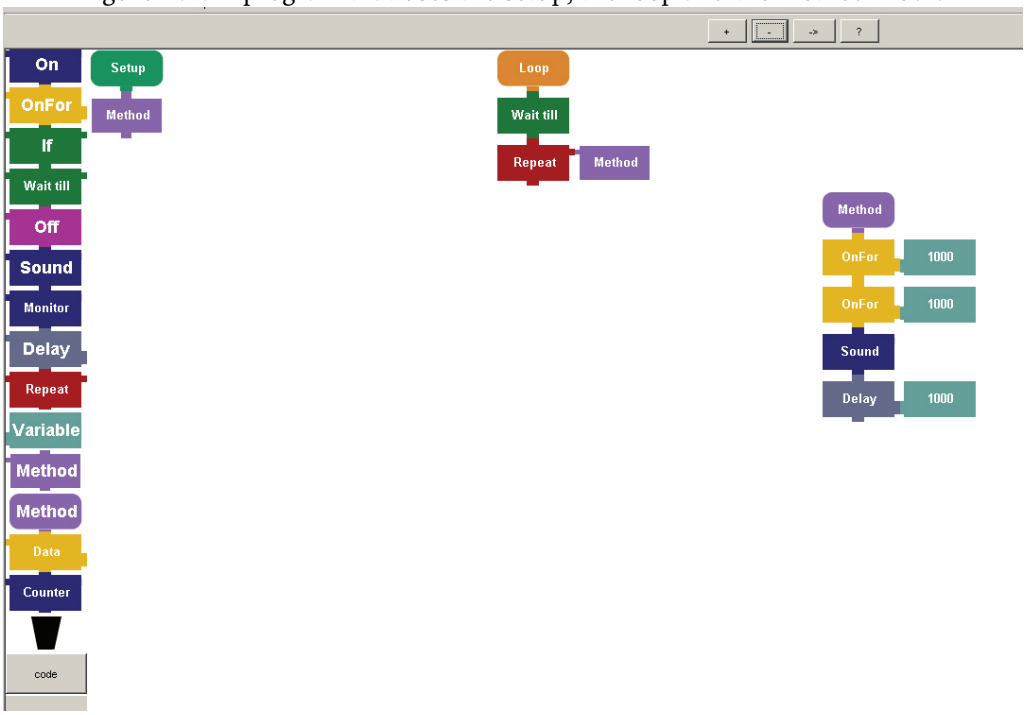
A.1 Loop, Setup and Method

The program (see figure A.1 for the visual representation of the program) will start by displaying a blink pattern once. Then it runs in a loop, waiting for a sensor value that is greater than a certain threshold (here: 344, see line 22 in listing A.1). Whenever this event occurs the program will display the blink pattern five times and then ask the sensor again.

In the visual representation (cf. figure A.1), you can see the frame with all blocks to choose from, on the left. Right of this frame you can see the block called "Setup". Underneath, there is a angled block called "Method". When the program starts (e. g. by restarting the micro-controller) it will execute the setup block and what is underneath: The method block calls a method (namely "blink_pattern"). The method can be everywhere on the canvas (here: on the right of the canvas), the block called "Method" that has rounded edges indicates that a method starts. All blocks directly under the method block (or right connected to a suitable block) are interpreted as belonging to the method. In the example that are: an onfor block with a connected variable block, then another onfor block with a connected variable block, a sound block and finally a delay block (that pauses the program for the duration of the value of the variable) with a connected variable block.

After executing the setup method and all blocks underneath it, the program will run into the loop (and execute every block under the loop block). That is, in our example, the waituntil block and then, when the condition (here: the value we read from pin 0 is greater than 344—see line 22 in listing A.1) is true, goes on to the next block (labeled "Repeat"). The repeat block is right-connected to the method block and will call a method (once again the blink_pattern method) five times (see line 25 till line 28 in listing A.1).

Figure A.1: A program that uses the setup, the loop and the method block.



Listing A.1: Arduino code

```
#include <Melody.h>

int val= 0;
int compare=0;
int Pin0 = 0;
int xo;int Variable110=1000;
int Pin4 = 4;
int Variable135=1000;
int Pin7 = 7;
int Pin9 = 9;
Melody myMelody = Melody(9);
int Variable185=1000;

void setup(){
pinMode(Pin7, OUTPUT);
pinMode(Pin0, INPUT);
pinMode(Pin4, OUTPUT);
myMelody.init();
blink_pattern();
}
void loop(){
```

```
22 while(analogRead(Pin0)>=344){
23 delay(10);
24 }
25 xo=0;
26 while (xo < 5){
27 blink pattern();
28 xo++;}
29
30 }
31
32 void blink_pattern()
33 {
34 digitalWrite(Pin4, HIGH);
35 delay(Variable110);
36 digitalWrite(Pin4, LOW);
37 delay(Variable110);
38 digitalWrite(Pin7, HIGH);
39 delay(Variable135);
40 digitalWrite(Pin7, LOW);
41 delay(Variable135);
42 myMelody.playTone('c',16);
43 delay(100);
44 delay(Variable185);
45 }
```

A.2 Jacket for the Blind

Another example of a project is from the smart fashion workshop. Two boys made a jacket that was meant to facilitate life of blind people (see figure A.2 for a visual impression of the jacket). The jacket had a light sensor attached; when it was dark in the environment an LED on the jacket would light up and a small vibration motor would vibrate to tell the wearer that her jacket just detected darkness. The project had one sensor—the light sensor. As actuators, it had the LED as well as the vibration motor.

The program is short and typical for the workshops. Figure A.3 shows the program in its visual representation. All blocks are placed under the loop block and will be executed in every loop. The first block under "Loop" is "Monitor". With the monitor block, users can see values of the serial port on their computer. When you look at the source code in listing A.2, you can see that in the loop a local variable called "analogueValue" is declared (line 14) and that the value of pin 5 is read into the variable (line 15). The value of the variable is then printed to serial (line 16). This is what the monitor block represents.

Listing A.2: Arduino code of the jacket for the blind

```
1 int val= 0;
```

Figure A.2: Presenting the jacket for the blind on the catwalk

```
2  int compare=0;
3  int Pin13 = 13;
4  int Pin5 = 5;
5  int Pin6 = 6;
6  void setup(){
7  pinMode(Pin5, INPUT);
8  pinMode(Pin13, OUTPUT);
9  pinMode(Pin6, OUTPUT);
10 Serial.begin(9600);
11
12 }
13 void loop(){
14 int analogValue;
```

Figure A.3: The jacket for the blind's program in its visual representation


```
15 analogValue = analogRead(5);
16 Serial.println(analogValue);
17 delay(10);
18 val = analogRead(Pin5);
19 compare=980;
20 if (val > compare){
21 digitalWrite(Pin13, HIGH);
22
23 } else {
24 //add code for actuator here
25  }
26 val = analogRead(Pin5);
27 compare=980;
28 if (val < compare){
29 digitalWrite(Pin13, LOW);
30
31 } else {
```

```
32 //add code for actuator here
33  }
34 val = analogRead(Pin5);
35 compare=980;
36 if (val == compare){
37 digitalWrite(Pin6, HIGH);
38
39 } else {
40 //add code for actuator here
41  }
42 val = analogRead(Pin5);
43 compare=980;
44 if (val < compare){
45 digitalWrite(Pin6, LOW);
46
47 } else {
48 //add code for actuator here
49  }
50 }
51 }
```

Underneath the monitor you can see a block labeled “If”. The if block is connected to a right neighbor—an on block. In this case, the program will execute everything on the right of the if block before it goes on to the next block underneath—another if block.

In the source code in listing A.2, you can see that a global value called “compare” is set (in this case to 980) and that afterwards an if statement compares “compare” to a value it reads from pin 5 (pins and threshold were entered by the users when dragging the if block to the canvas). If the value read is greater than compare, pin 13 is set to high (that is what the on block does). In the boys’ project this construct is responsible for switching on the LED on the jacket when it is dark.

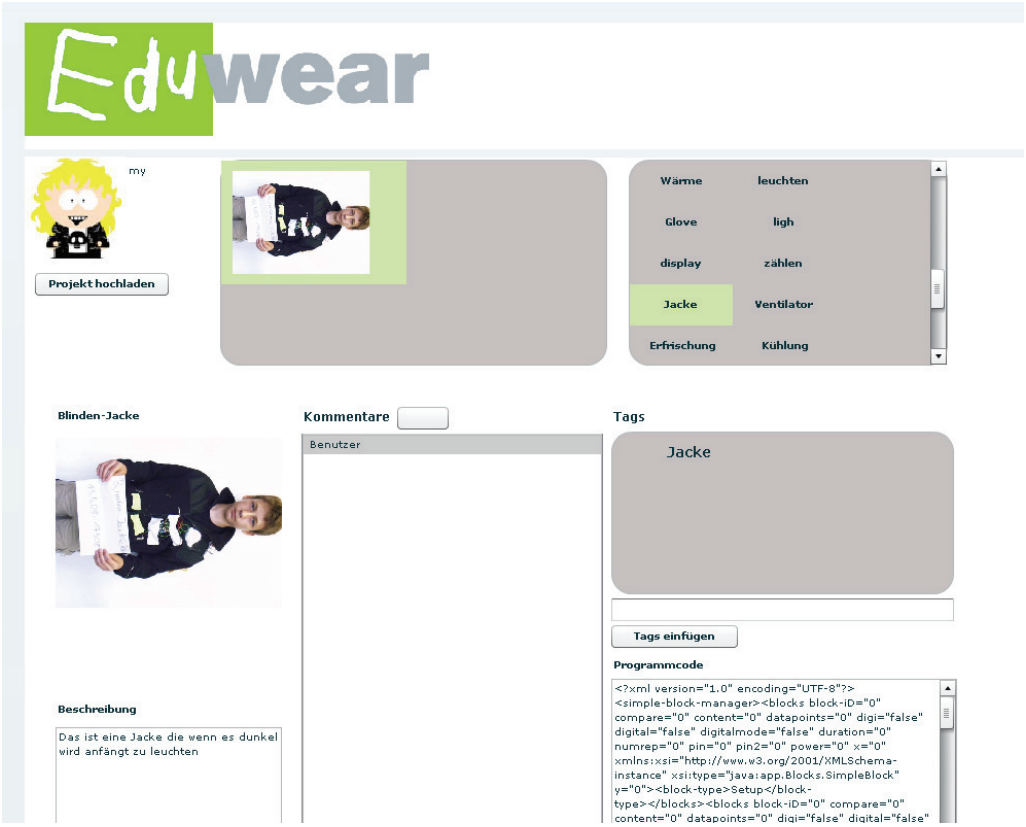
The second if block under loop in figure A.3 is responsible for asking for the value of the sensor connected to pin 5 again. When its value is less than “compare” the block sets pin 13 to low. This construct could also be an “else” in the first if-statement, but because there is no else block (to keep the number of blocks as low as possible), users solve the problem with using two if blocks asking the sensor for its values twice.

The next two if blocks underneath do a similar thing, but are responsible for switching on the vibration motor. Instead of switching it on, when the sensor value is greater than the threshold, they only switch it on, when the sensor equals the threshold to have only one short signal.

See figure A.4 for a screenshot of the project uploaded to Amici’s browser-based client. In the top left corner, you can see the currently logged-in user (which is the author). In the top middle, you see a list of projects for the currently selected tag (the tag “Jacke” is selected, there is only one project tagged “Jacke” in the database that is why only one project is shown here). In the top right frame, you see the folksonomy in

the system (“Jacke” is highlighted because it is selected). Underneath, you can see details for the selected project called “Blinden-Jacke”. On the left, there is the project’s picture and description. In the middle, you can see the comment box and on the right the source code in XML notation.

Figure A.4: The jacket for the blind’s program in Amici’s browser-based client



Appendix B

Questionnaires and Interview Guidelines

B.1 Questionnaires from the EduWear Project

During the EduWear projects, several questionnaires were used for evaluation: questionnaires for the participants, but as well for the teachers and tutors. In this thesis, we drew on some results of the questionnaires e. g. when writing about the function clothing had for children and young people or when describing learning experiences during the workshops. Therefore, you can find the questionnaires for participants for before as well as after the courses in the following part.

B.1.1 Questionnaires Before the Workshop

Workshop Title:

Date:

EduWear Evaluation Questionnaire for Participants
(Before Course)

Dear participant,
welcome to the EduWear workshop!

With this questionnaire we want to find out if our workshops are
in tune with your interests and expectations.

We are interested in your opinion. There are no “right” and “wrong” answers.

All information in the questionnaires will be treated anonymously,
That is, nobody will know what you have answered.

The participation in this survey is optional!
Your answers help us to improve the workshops.

Thank you!

We need a codenumber of each participant.
Please create your code number now, by filling into the boxes:

The last digit of your telephone number
The second letter of your mother's first name
The third letter of your own first name
The first letter of the street where you live

1. What made you come to the EduWear workshop?

		strongly agree	**agree**	**disagree**	**strongly disagree**
1.1	My parents sent me to this workshop				
1.2	Our teacher sent the class to this workshop				
1.3	I'm here because a friend is here too				
1.4	The topic of the workshop sounded interesting				

1.5. If it was the topic, what interested you in particular?

_____

_____

_____

_____

_____

2. What do you want to do during this workshop?

		strongly agree	**agree**	**disagree**	**strongly disagree**
2.1	I want to work with textiles				
2.2	I want to learn how to program				
2.3	I want to learn more about Smart Textiles (e.g. "intelligent" fabric, conductive yarn...)				
2.4	I want to design something beautiful				
2.5	I want to write my own small programs				
2.6	I want to learn something about the body and about movement				
2.7	I want to work in a group				
2.8	I want to solve technical problems				

3. What have you done often?

I have often...

		strongly agree	**agree**	**disagree**	**strongly disagree**
3.1	Occupied myself with technical construction kits or tinkered with technical things (e.g. an electric bell)				
3.2	Worked with textiles at school				
3.3	Worked with textiles at home				
3.4	Written simple programs				
3.5	Opened up a computer and installed something				
3.6	Opened up a technical device (radio, clock etc.) to see how it works				

		strongly agree	**agree**	**disagree**	**strongly disagree**
3.7	Read about how technical devices work				
3.8	Talked with friends about computers and technology				

4. Do the following statements apply to you?

		strongly agree	**agree**	**disagree**	**strongly disagree**
4.1	When others talk about computers, I can join in				
4.2	The computer usually does what I want it to do				
4.3	When others talk about technology, I can join in				
4.4	If I wanted to, I could become a computer expert				
4.5	If I wanted to, I could program my own computer game				
4.6	If I put in some effort, I can be good at technology and handicrafts				
4.7	Working with textiles is fun				
4.8	Learning how to sew is unnecessary				

		strongly agree	**agree**	**disagree**	**strongly disagree**
4.9	To me, most high technology seems like "magic"				
4.10	Only scientists know how high technology works				

5. Your wishes and plans

		strongly agree	**agree**	**disagree**	**strongly disagree**
5.1	I'd like to have more classes that deal with technology				
5.2	I'd like to have more classes that deal with Smart Textiles				
5.3	I could imagine working with computers when I finish school				
5.4	In the future I could imagine working in a profession that has to do with fashion or design				

6. What does clothing mean to you?

For me, clothes are...

		strongly agree	**agree**	**disagree**	**strongly disagree**
6.1	like a second skin				
6.2	protection				
6.3	not really that important				
6.4	a means of expressing myself				
6.5	a means of setting myself apart from others				
6.6	a means of showing what group(s) I belong to or who my friends are				
6.7	something that has nothing to do with electronics				

6.8 Other: _____

7. Where can you work on a computer?

7.1. We have a computer at home.

7.2. I have my own computer.

7.3. I often work on a computer at school.

8. Did you participate in another technology-workshop before this one?

8.1. I participated in a robotics-workshop.

8.2. I participated in a Smart Textiles workshop.

9. Personal Information

Age: I am ____ years old and I go to ___ grade.

I am a girl boy

What language(s) do you speak at home? _____

Thank you for filling out this questionnaire!

B.1.2 Questionnaires After the Workshop

Workshop Title:

Date:

EduWear Evaluation Questionnaire for Participants
(After Course)

Last updated 26th of July 2007

Dear participant,
now we will ask you for your opinion again.

We are interested in your opinion. There are no „right“ and „wrong“ answers.

All information in the questionnaires will be treated anonymously,
that is, nobody will know what you have answered.

The participation in this survey is optional!
Your answers help us to improve the workshops.

Thank you!

Please create your code number again, by filling into the boxes:

The last digit of your telephone number
The second letter of your mother's first name
The third letter of your own first name
The first letter of the street where you live

1. How did you like the workshop?

		strongly agree	**agree**	**disagree**	**strongly disagree**
1.1	Participating in the workshop was fun				
1.2	I would recommend such a workshop to a friend				
1.3	I would like to participate in a follow-up workshop				
1.4	Maybe I will ask for a Smart Textiles toolkit for my birthday				

2. How do the following statements apply to you?

		strongly agree	**agree**	**disagree**	**strongly disagree**
2.1	I learned something new about Smart Textiles				
2.2	I learned something new about electronics				
2.3	I learned something new about programming				
2.4	I would have needed more support				
2.5	I would have liked to do more programming				
2.6	I would have liked to work more with the textile material				

		strongly agree	**agree**	**disagree**	**strongly disagree**
2.7	I would have liked to learn more about electronics				
2.8	I would have liked to work alone more often				
2.9	The workshop should have been longer				
2.10	I was able to realize my own ideas				
2.11	The tasks were too limited				
2.12	The tutors were able to help our group when we had problems				

3. What kind of Smart Textile-artefact did you build in this workshop?

Please draw or describe your product:

4. Do the following statements apply to you?

		strongly agree	**agree**	**disagree**	**strongly disagree**
4.1	When others talk about computers, I can join in				
4.2	The computer usually does what I want it to do				
4.3	When others talk about technology, I can join in				
4.4	If I wanted to, I could become a computer expert				
4.5	If I wanted to, I could program my own computer game				
4.6	If I put in some effort, I can be good at technology and handicrafts				
4.7	Working with textiles is fun				
4.8	Learning how to sew is unnecessary				
4.9	To me, most high technology seems like "magic"				
4.10	Only scientists know how high technology works				

5. Your wishes and plans

		strongly agree	**agree**	**disagree**	**strongly disagree**
5.1	I'd like to have more classes that deal with technology				
5.2	I'd like to have more classes that deal with Smart Textiles				
5.3	I could imagine working with computers when I finish school				
5.4	In the future I could imagine working in a profession that has to do with fashion or design				

6. What does clothing mean to you?
For me, clothes are...

		strongly agree	**agree**	**disagree**	**strongly disagree**
6.1	like a second skin				
6.2	protection				
6.3	not really that important				
6.4	a means of expressing myself				
6.5	a means of setting myself apart from others				

		strongly agree	**agree**	**disagree**	**strongly disagree**
6.6	a means of showoing what group(s) I belong to or who my friends are				
6.7	something that has nothing to do with electronics				

6.8 Other: _____

Thank you for filling out this questionnaire!

B.2 Guidelines Tagging Interview

1. Introduction: Explanations what the goals of the interviews are, how long the interview was supposed to last.
2. Do you remember that you uploaded your projects? (supported by showing the software and some projects on the smart board)
3. You had to add some words for description. Tags. Why are they important?
4. Let's look at one project from you (show project on the smart board).
5. Why did you tag it that way?
6. (show tooltips on the blocks)
7. Why did you tag it that way?
8. Do you find the tags useful at this place?